Transaction Processing Facility

Operations

Version 4 Release 1

SH31-0162-15

Transaction Processing Facility

Operations

Version 4 Release 1

SH31-0162-15

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page xxi.

Sixteenth Edition (June 2002)

This is a major revision of, and obsoletes, SH31-0162-14 and all associated technical newsletters.

This edition applies to Version 4 Release 1 Modification Level 0 of IBM Transaction Processing Facility, program number 5748-T14, and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

IBM welcomes your comments. Address your comments to:

IBM Corporation
TPF Systems Information Development
Mail Station P923
2455 South Road
Poughkeepsie, NY 12601-5400
USA

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Notices

References in this book to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service in this book is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 830A
Mail Drop P131
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Any pointers in this book to non-IBM Web sites are provided for convenience only and do not in any way serve as an endorsement. IBM accepts no responsibility for the content or use of non-IBM Web sites specifically mentioned in this book or accessed through an IBM Web site that is mentioned in this book.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Peer-to-Peer Networking
APPN
C/370
DB2
ECKD
Enterprise Systems Connection Architecture
ESCON
IBM
MQSeries
Processor Resource/Systems Manager
PR/SM

S/370
Sysplex Timer
System/390
VisualAge
VTAM
3090.

Other company, product, and service names may be trademarks or service marks of others.

About This Book

TPF Operations contains detailed descriptions for computer room operations and control program management.

TPF Network Extension Facility (NEF) support depends on either:
- The Network Extension Facility (NEF2 PRPQ P85025), or
- Airlines Line Control Interface (ALCI) feature of ACF/NCP.

All subsequent occurrences of NEF in the text of this book are superseded by the previous information.

TPF Network Control Program (NCP) support depends on:
- Subarea support (T5), which requires SNA Network Interconnection (SNI)
- T2.1 low-entry networking (LEN) support, which requires NCP Version 4 Release 3, or later
- T2.1 Advanced Peer-to-Peer Networking (APPN) support, which requires NCP Version 6 Release 2, or later.

All subsequent occurrences of NCP in the text of this book are superseded by the previous information, unless otherwise specified.

In this book, abbreviations are often used instead of spelled-out terms. Every term is spelled out at first mention followed by the all-caps abbreviation enclosed in parentheses; for example, Systems Network Architecture (SNA). Abbreviations are defined again at various intervals throughout the book. In addition, the majority of abbreviations and their definitions are listed in the master glossary in the *TPF Library Guide*.

Before You Begin

Before using this book, be sure you have a basic understanding of the TPF system. See *TPF Concepts and Structures* for a comprehensive technical overview of the TPF system.

Who Should Read This Book

TPF Operations is intended for:
- System console operators responsible for routine operations of the TPF system
- System programmers responsible for tuning the TPF system and advising the console operator in exception situations
- Functional console operators responsible for subsets of system activity, such as database or network management.

How This Book Is Organized

This book is divided into two main parts, as follows:
- Part 1, “Control Program Operations” contains an introduction to the TPF control program (including IPL procedures) and a reference guide for the commands that are used to operate the TPF system. The commands make up the bulk of this book and are organized in alphabetic order.

- Part 2, “System Utilities Operations” contains information about how to use the system utility packages that are available for the TPF system. Some of these utilities are controlled using commands. These commands are described in Part 1, “Control Program Operations”.

Conventions Used in the TPF Library

The TPF library uses the following conventions:

<table>
<tr><th>Conventions</th><th>Examples of Usage</th></tr>
<tr><td>*italic*</td><td>Used for important words and phrases. For example:
A *database* is a collection of data.

Used to represent variable information. For example:
Enter **ZFRST STATUS MODULE** *mod*, where *mod* is the module for which you want status.</td></tr>
<tr><td>**bold**</td><td>Used to represent text that you type. For example:
Enter **ZNALS HELP** to obtain help information for the ZNALS command.

Used to represent variable information in C language. For example:
level</td></tr>
<tr><td>`monospaced`</td><td>Used for messages and information that displays on a screen. For example:
`PROCESSING COMPLETED`

Used for C language functions. For example:
`maskc`

Used for examples. For example:
`maskc(MASKC_ENABLE, MASKC_IO);`</td></tr>
<tr><td>***bold italic***</td><td>Used for emphasis. For example:
You ***must*** type this command exactly as shown.</td></tr>
<tr><td>**Bold underscore**</td><td>Used to indicate the default in a list of options. For example:
Keyword=OPTION1 | **DEFAULT**</td></tr>
<tr><td>Vertical bar |</td><td>Used to separate options in a list. (Also referred to as the OR symbol.) For example:
Keyword=Option1 | **Option2**

Note: Sometimes the vertical bar is used as a *pipe* (which allows you to pass the output of one process as input to another process). The library information will clearly explain whenever the vertical bar is used for this reason.</td></tr>
<tr><td>CAPital LETters</td><td>Used to indicate valid abbreviations for keywords. For example:
KEYWord=*option*</td></tr>
<tr><td>Scale</td><td>Used to indicate the column location of input. The scale begins at column position 1. The plus sign (+) represents increments of 5 and the numerals represent increments of 10 on the scale. The first plus sign (+) represents column position 5; numeral 1 shows column position 10; numeral 2 shows column position 20 and so on. The following example shows the required text and column position for the image clear card.
`|...+....1....+....2....+....3....+....4....+....5....+....6....+....7...`

`LOADER   IMAGE CLEAR`

Notes:
1. The word LOADER must begin in column 1.
2. The word IMAGE must begin in column 10.
3. The word CLEAR must begin in column 16.</td></tr>
</table>

How to Read the Syntax Diagrams

This section describes how to read the syntax diagrams (informally called *railroad tracks*) used in this book.

- Read the diagrams from left-to-right, top-to-bottom, following the main path line. Each diagram begins on the left with double arrowheads and ends on the right with 2 arrowheads facing each other.

```
►►──┤ Syntax Diagram ├──────────────────────────────────────────────►◄
```

- If a diagram is longer than one line, the first line ends with a single arrowhead and the second line begins with a single arrowhead.

```
►►──┤ The first line is long and extends the width of the diagram ├──────────►

►─┤ Second Line ├──────────────────────────────────────────────────►◄
```

- A word in all uppercase is a parameter that you must spell ***exactly*** as shown.

```
►►──PARAMETER─────────────────────────────────────────────────────►◄
```

- If you can abbreviate a parameter, the optional part of the parameter is shown in lowercase. (You must type the text that is shown in uppercase. You can type none, one, or more of the letters that are shown in lowercase.)

  Note: Some TPF commands are case-sensitive and contain parameters that must be entered exactly as shown. This information is noted in the description of the appropriate commands.

```
►►──PARAMeter─────────────────────────────────────────────────────►◄
```

- A word in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

```
►►──variable─────────────────────────────────────────────────────────►◄
```

- Required parameters and variables are shown on the main path line. You must code required parameters and variables.

```
►►──REQUIRED_PARAMETER──required_variable────────────────────────────►◄
```

- If there is more than one mutually exclusive required parameter or variable to choose from, they are stacked vertically.

```
►►──┬─REQUIRED_PARAMETER_1─┬─────────────────────────────────────────►◄
    ├─REQUIRED_PARAMETER_2─┤
    ├─required_variable_a──┤
    └─required_variable_b──┘
```

- Optional parameters and variables are shown below the main path line. You can choose not to code optional parameters and variables.

```
►►──┬─────────────────────────────────────────┬──────────────────────►◄
    └─OPTIONAL_PARAMETER──optional_variable──┘
```

- If there is more than one mutually exclusive optional parameter or variable to choose from, they are stacked vertically below the main path line.

```
►►──┬──────────────────────┬─────────────────────────────────────────►◄
    ├─OPTIONAL_PARAMETER_1─┤
    ├─OPTIONAL_PARAMETER_2─┤
    ├─optional_variable_a──┤
    └─optional_variable_b──┘
```

- An arrow returning to the left above a parameter or variable on the main path line means that the parameter or variable can be repeated. The comma (,) means that each parameter or variable must be separated from the next parameter or variable by a comma.

```
   ┌─,───────────────────┐ ┌─,──────────────────┐
   ▼                     │ ▼                    │
►►───REPEATABLE_PARAMETER─┴───repeatable_variable─┴──────────────────────────►◄
```

- An arrow returning to the left above a group of parameters or variables means that more than one can be selected, or a single one can be repeated.

```
   ┌─,────────────────────────┐
   ▼                          │
►►───┬─REPEATABLE_PARAMETER_1─┬┴──────────────────────────────────────────►◄
     ├─REPEATABLE_PARAMETER_2─┤
     └─repeatable_variable────┘
```

- If a diagram shows a blank space, you must code the blank space as part of the syntax. In the following example, you must code **PARAMETER** *variable*.

```
►►──PARAMETER── ──variable───────────────────────────────────────────────►◄
```

- If a diagram shows a character that is not alphanumeric (such as commas, parentheses, periods, and equal signs), you must code the character as part of the syntax. In the following example, you must code **PARAMETER=(***begin***.***end***)**.

```
►►──PARAMETER=(begin.end)────────────────────────────────────────────────►◄
```

- Default parameters and values are shown above the main path line. The TPF system uses the default if you omit the parameter or value entirely.

```
   ┌─DEFAULT───┐ ┌─0────────┐
►►─┴─PARAMETER─┴─┴─variable─┴────────────────────────────────────────────►◄
```

- References to syntax notes are shown as numbers enclosed in parentheses above the line. Do not code the parentheses or the number.

```
             (1)
▶▶──PARAMETER──────────────────────────────────────────────────────▶◀
```

Notes:

1 An example of a syntax note.

- Some diagrams contain *syntax fragments*, which serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names are in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.

```
▶▶──┤ Reference to Syntax Fragment ├───────────────────────────────▶◀
```

Syntax Fragment:

```
├──1ST_PARAMETER,2ND_PARAMETER,3RD_PARAMETER───────────────────────┤
```

Related Information

A list of related information follows. For information on how to order or access any of this information, call your IBM representative.

IBM Transaction Processing Facility (TPF) 4.1 Books

- *TPF ACF/SNA Data Communications Reference*, SH31-0168
- *TPF ACF/SNA Network Generation*, SH31-0131
- *TPF Application Programming*, SH31-0132
- *TPF Application Requester User's Guide*, SH31-0133
- *TPF C/C++ Language Support User's Guide*, SH31-0121
- *TPF Concepts and Structures*, GH31-0139
- *TPF Database Reference*, SH31-0143
- *TPF Data Communications Services Reference*, SH31-0145
- *TPF General Macros*, SH31-0152
- *TPF Main Supervisor Reference*, SH31-0159
- *TPF Migration Guide: Program Update Tapes*, GH31-0187
- *TPF Migration Guide: TPF 3.1 System to TPF 4.1 System*, GH31-0186
- *TPF Multi-Processor Interconnect Facility Reference*, SH31-0155
- *TPF Non-SNA Data Communications Reference*, SH31-0161
- *TPF Program Development Support Reference*, SH31-0164
- *TPF System Generation*, SH31-0171
- *TPF System Installation Support Reference*, SH31-0149
- *TPF System Macros*, SH31-0151
- *TPF System Performance and Measurement Reference*, SH31-0170

- *TPF Transmission Control Protocol/Internet Protocol*, SH31-0120.

IBM Message Queuing Books

- *MQSeries Command Reference*, SC33-1369
- *MQSeries Distributed Queue Management Guide*, SC33-1139.

Miscellaneous IBM Books

- *Character Data Representation Architecture Reference and Registry*, SC09-2190
- *ESA/390 Principles of Operation*, SA22-7201
- *3704/3705 Control Program Generation Utilities Guide*, GC30-3008
- *3880 Storage Control Record Cache RPQ Description*, GA32-0087
- *3880 Storage Control Record Cache RPQ Introduction*, GA32-0086
- *3990 Storage Control Operations and Recovery Guide*, GA32-0253
- *3990 Storage Control Planning, Installation, and Storage Administration*, GA32-0100
- *3990 Transaction Processing Facility Support RPQs*, GA32-0134.

Non-IBM Books

- *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* from the POSIX series of standards for applications and user interfaces to open systems, copyrighted by the Institute of Electrical and Electronics Engineers (IEEE).

Online Information

| - *Messages (Online)*
| - *Messages (System Error and Offline)*
- *SSL for the TPF 4.1 System: An Online User's Guide*
- *VisualAge TPF Online Help*.

How to Send Your Comments

Your feedback is important in helping to provide the most accurate and highest quality information. If you have any comments about this book or any other TPF information, use one of the methods that follow. Make sure you include the title and number of the book, the version of your product and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

- If you prefer to send your comments electronically, do either of the following:
  - Go to http://www.ibm.com/tpf/pubs/tpfpubs.htm.

    There you will find a link to a feedback page where you can enter and submit comments.
  - Send your comments by e-mail to tpfid@us.ibm.com
- If you prefer to send your comments by mail, address your comments to:

  IBM Corporation
  TPF Systems Information Development

Mail Station P923
2455 South Road
Poughkeepsie, NY 12601-5400
USA

- If you prefer to send your comments by FAX, use this number:
  - United States and Canada: 1 + 845 + 432 + 9788
  - Other countries: (international code) + 845 + 432 +9788

Part 1. Control Program Operations

Introduction to TPF Operations

The TPF control program (CP) is main storage resident and cannot be directly controlled by a console operator. However, a number of file resident programs are available that allow you to communicate with the control program and direct its activities. Other programs are available for hardware manipulation. All of these file resident programs are collectively called command programs.

A certain amount of key data resides in the control program that is critical to system operation. This data is necessary during restart to ensure continuity of processing. Therefore, all changes are saved in file records known as keypoint records. During restart, these records are used to reinitialize the control program to its former state.

There are three phases associated with system initialization. The first phase concerns the initial loading of programs (IPL); the second phase concerns the startup or restart of the control program; the third phase concerns the reloading of programs if a catastrophic error occurs.

Initial Load

The TPF system is loaded to the online system in 2 steps:

1. The online keypoints and the file resident and main storage resident program segments are loaded to the loader general file by the system loader offline segment. See *TPF System Installation Support Reference* for more information.
2. The online keypoints and the file resident and main storage resident program segments are loaded from the loader general file to the online files by IPLing the loader general file.

Note: The loader general file and the online files must be initialized by ICKDSF, an MVS utility, to include the IPL program and be formatted with the real-time disk formatter (FMTR). See *TPF Database Reference* for more information about FMTR.

The TPF system is started by IPLing the prime (or backup) online system pack. See *TPF Main Supervisor Reference* for more information about the initial program load (IPL).

Loosely Coupled IPL Procedure

All of the processors in a loosely coupled complex are IPLed from the same online files. The first IPLed processor sets the time-of-day (TOD) clock value for the complex. Thereafter, when the remaining processors are IPLed, the TOD clocks are synchronized with the TOD clock of the first processor or with the Sysplex Timer, if it is available. Installations that use the multiple database function (MDBF) configure all subsequent processors from 2 sources: processor-unique keypoints and the MDBF configuration supplied by the first processor through shared keypoints. To IPL the first processor in a complex:

1. Perform a hardware IPL for the processor.
2. When you are prompted, enter the TPF logical CPU ID of the processor that is being IPLed. TPF logical processor IDs are chosen by the system programmer during the system generation process.
3. When you are prompted, specify the type of IPL that you want to perform.
4. If you specified an IPL with image selection, enter the name of the TPF image when you are prompted.

5. When you are prompted, enter the MDBF configuration information for the complex (if MDBF was generated in the system).

   The system is now cycled to 1052 state.
6. Set the system interprocessor status variables using the ZSIPC command. The Multi-Processor Interconnect Facility (MPIF) system and complex variables must be set and at least one device, path class, and path must be defined using the ZMPIF command.

   Note: It is recommended that you dedicate a path class to MPIF IPC and that you define this path class as a protected class.
7. Perform either a software or hardware IPL. A software IPL will be completed without operator interaction.

   The system is placed in 1052 state. Other processors can now be added to the complex.

After the first processor is IPLed, all the other processors can be added to the complex using the following procedure.
1. Perform a hardware IPL for the processor.
2. When you are prompted, enter the TPF logical CPU ID of the TPF system that you want to IPL.

   Note: Only one physical CPU can have a particular TPF logical CPU ID at any one time. When adding a processor to a complex, use a logical ID that is not already active in the complex.
3. The MPIF system name must be set and at least one path must be defined using the ZMPIF command. Perform either a software or hardware IPL. A software IPL will be completed without operator interaction.
4. The IPL procedure will recognize that other processors in the complex exist and use their MDBF configuration if MDBF was generated in the system.
5. While the system is in restart mode, the system will force a software-initiated IPL of the processor to synchronize the processor-shared tables with those of the first processor. This process is completed without operator intervention.

   The IPL is complete when the system is placed in 1052 state.

If 2 or more TPF processors are IPLed at the same time, each one will be unaware of the other until they go through MPIF restart.

If the TPF systems that are being IPLed are currently active in the loosely coupled complex, the IPLs will continue correctly, generally without operator intervention.

If the TPF systems that are being IPLed are not currently active, but are joining a currently active TPF loosely coupled complex, they will go through a forced software IPL that will bring them into the complex. The software IPLs will ensure that the TPF processor ID you specified is unique and that all keypoint data is current. This forced IPL is the normal mechanism for adding processors to an active TPF loosely coupled complex. (See the previous procedure for more information.)

If there is no currently active loosely coupled complex (all TPF processors are deactivated), each system that is being IPLed will perform as if it is the first TPF system to be IPLed. If more than one TPF system performs as if it is the first system being IPLed, confusing and occasionally contradictory messages may be displayed on the consoles of the system being IPLed. With MDBF, each TPF system will request the MDBF configuration, possibly getting different responses from you. With the Record Cache RPQ, each TPF system will attempt to reinitialize the record cache and reset the record cache operational parameters, possibly causing I/O errors on the other systems attempting to IPL.

There are a series of checks during system restart to find keypoint conditions that are out of sync. These conditions can cause database damage. The checks will also show if more than one processor determined that it is the first processor in the complex, and will remove all but one of the processors that are being IPLed. These processors can then be brought into the loosely coupled complex by performing an additional IPL.

When creating a TPF loosely coupled complex (for example, IPLing the first TPF processor), do not IPL additional processors until the first processor is in 1052 state. Once the TPF complex is created, new processors can be added at any time.

Note: Adding TPF processors to a loosely coupled complex places an I/O load against the module that is being IPLed. Installations that are very close to the DASD I/O access rate limit on the IPL device should IPL sequentially to reduce device contention.

IPL Program Wait States

The following information lists and explains the program status words (PSWs) for the IPL program wait states.

PSW	Explanation
000A0000 000003FF	Loaded by IPLB at the normal end of dump processing.
000A0000 000000FB	Loaded by IPLB when a CC3 is returned from a set clock (SCK) instruction indicating that the TOD clock is not operational.
000A0000 0000DEAD	Loaded by IPLB when an irrecoverable I/O error is found during dump processing.
000A0000 00DEAD00	Loaded by IPL2, IPLA, and IPLB when an unexpected machine check interrupt is found.
000A0000 00FFFFF0	Loaded by IPL2, IPLA, and IPLB when an unexpected external interrupt is found.
000A0000 00FFFFF2	Loaded by IPL2, IPLA, and IPLB when an unexpected supervisor call interrupt is found.
000A0000 00FFFFF4	Loaded by IPL2, IPLA, and IPLB when an unexpected program check interrupt is found.
000A0000 00FFFFF8	Loaded by IPLA and IPLB when an unexpected I/O interrupt is found. It is loaded by IPL2 when an irrecoverable error, such as CC3 on a start subchannel (SSCH) or test subchannel (TSCH), is found before the I/O interrupt.
000A0000 00*xxxx*F6	Loaded by IPL2 when an incorrect record ID is found in the IPLA chain, where *xxxx* is the ID of the incorrect record.
000A0000 00*xxxx*F8	Loaded by IPL2 when an irrecoverable I/O error is detected, where *xxxx* is the cumulative device and subchannel status from the failing device.

000A0000 FFFFFFFA	Loaded by IPLB (IB01) when storage for the prefix pages, common I/O (CIO), or the CIO work areas cannot be allocated without overlaying something that was loaded by IPLA.
000A0000 FFFFFFFC	Loaded by IPLB (IB01) when fast recovery objects overlap.
000A0000 FFFFFFFE	Loaded by IPLB (IB01) when storage for the IPL restart area, the segment table, or the page tables cannot be allocated without overlaying common I/O (CIO).
000A0000 *xxxxxxxx*	Loaded by IPLB when a program calls the dump routine a second time while a dump is in progress. The second call can be a result of a program check or a branch from an IPLB error routine. If a program check occurred, the program interruption code in the prefix page indicates the type of exception. The addresses of the page and segment tables and the register save area are contained in the IPL restart area, where *xxxxxxxx* is the address of the IPL restart area (defined by the IB5CT DSECT in IB0CT). The page and segment tables and the register save area can be dumped using the stand-alone dump (SADUMP) utility.

System Initialization (CCCTIN) Wait States

The following information lists and explains the PSWs for the system initialization wait states.

PSW	Explanation
040AC000 00FFFFFF	Loaded by CCCTIN to indicate the normal end of the CCCTIN micro dump or to indicate that an irrecoverable error was detected while attempting to write to the system console.
000A0000 C3E3F8F5	Loaded by CT85 when an application I-stream (for example, the MPIF I-stream) fails to restart. IPL the TPF system to recover.

System Error Processing Program Wait States

The following information lists and explains the PSWs for the system error program wait states. The low-order word of each disabled wait PSW loaded by CPSE does not address an instruction but, instead, contains a unique 3-byte identifier:

PSW	Explanation
040AC000 00CCD01A	Catastrophic error on a general file (GF) IPL. A dump was taken, but no recovery is attempted.
000AC000 00CCD01E	System virtual memory (SVM) is not stable; unable to switch from real mode to home address space. Inspect the control registers.
000AC000 00CCD01F	CCCPSF recursion failed; unable to continue.
040AC000 00CCD09E	Program error in CPSE on an application I-stream.

000A0000 00DEAD09	An unexpected first-level interrupt handler (FLIH) machine check occurred.
040A0000 00CCD708	CFLF restart failed on a processor other than the first in the complex.

Machine Check Interruption Handling Disabled Wait States

The following information lists and describes the disabled wait states related to the machine check interruption handler. The low-order 3 bytes of each PSW contains an identifier.

Note: Recover from these disabled wait states by performing an IPL with the CLEAR option.

PSW	Explanation
000A0000 00DEAD01	The machine check FLIH was interrupted twice by machine check interruptions (for example, double machine check recursion).
000A0000 00DEAD02	A machine check interruption reported channel subsystem damage.
000A0000 00DEAD03	A machine check interruption on an application I-stream reported catastrophic damage while system error recovery was active on the main I-stream, or a check-stop condition was reported to an application I-stream while system error recovery was active on the main I-stream.
000A0000 00DEAD04	While handling a machine check interruption that was reporting catastrophic damage, the machine check FLIH could not perform a SIGP STOP for one of the I-streams; or while handling a check-stop condition, the check-stop handler was unable to perform a SIGP STOP for one of the I-streams.
000A0000 00DEAD05	While handling a machine check interruption that was reporting catastrophic damage, the machine check FLIH could not perform a SIGP SENSE for one of the I-streams; or while handling a check-stop condition, the check-stop handler could not perform a SIGP SENSE for one of the I-streams.
000A0000 00DEAD06	While handling a machine check interruption that was reporting catastrophic damage, the machine check FLIH determined that the main I-stream had performed a check stop; or while handling a check-stop condition, the check-stop handler determined that the main I-stream had performed a check stop.
000A0000 00DEAD07	While handling a machine check interruption that was reporting catastrophic damage, the machine check FLIH could not perform a SIGP RESTART for the main I-stream; or while handling a check-stop condition, the check-stop handler could not perform a SIGP RESTART for the main I-stream.
000A0000 00DEAD08	While handling a machine check interruption that

	was reporting catastrophic damage, the machine check FLIH was interrupted by a machine interruption on the main I-stream.
000A0000 00DEAD09	An unexpected FLIH machine check occurred.

Restart

The control program can be restarted by performing a hardware IPL or by issuing the ZRIPL command. Whenever you restart the control program, an initial copy of the control program is loaded from file to main storage. The keypoint records are then loaded from the IPLed file and are used to initialize various fields and tables in main storage. All working storage is initialized with all main storage blocks on the uncommitted storage lists. During initialization, a check is performed to determine whether a dump tape is mounted and ready. If one is not mounted, you are notified and no more action occurs until the you mount a tape. When initialization is completed, the following message is displayed on the system console:

`CVRN0004I` *hh.mm.ss* `RESTART COMPLETED- 1052 STATE`

Note: Time stamping is performed on most control program output messages. Where applicable, *hh.mm.ss* indicates time in hours, minutes, and seconds. However, you should realize that the software clocks are not running until the system is cycled above 1052 state. As a result, the time stamp of some messages is the same as the time that the software clocks were last active.

At this point, the system accepts input only from the system console or local 3270 terminals. You can perform functions such as modifying polling status, mounting tapes, setting time, and modifying file records.

Reload

If it is necessary to reload the TPF system from the load module library, you have the option of either reloading the keypoint records with an initial copy or leaving them in their updated state. This is determined by a control card option, which is explained in *TPF System Installation Support Reference*.

Machine States

The system can operate in any one of the following states:
- 1052 state
- Utility (UTIL) state
- CRAS state
- Message switching state
- Normal (NORM) state.

To change from one state to another, enter the ZCYCL command.

1052 State

1052 state is defined as follows:

- Only Z action codes are accepted from the system console or 3270 local devices that are logged on to the system message processor. See *TPF Data Communications Services Reference* for more information about the system message processor.
- The software clocks are not stepping.

- The time available supervisor (TAS) is not active.
- The get file storage (GFS) facility is not active. An attempt to issue a GETSC, GETLC, GETFC, or RELFC macro results in a system error.
- Keypoint update is active.
- There is no input or output line activity, except for 3270 local terminals in the CRAS table that are logged on to the system message processor.
- All SEND and CRAS macros are routed to the system console regardless of the line number, except for 3270 local output. Multiple segment messages are not allowed and, if found, the remaining segments will be lost.
- The long message transmitter and output message transmitter may still have output messages in queue, but these queues are not being serviced. See *TPF Data Communications Services Reference* and *TPF ACF/SNA Data Communications Reference* for more information about these programs.
- Interrupts are accepted from 3270 local devices, but messages will be rejected unless the terminal is logged on to the system message processor or it is a log in message to the system message processor. The device must also be listed in the CRAS table for valid input. Multisegment (long) input messages and certain LOGU and LOGP functions will also be rejected because they use pool records. Incorrect input will result in the following message:

  ```
  SYSTEM RESTRICTED, RETRY IN 5 MIN.
  ```
- STIMC time-initiated requests are accepted and dispatched. STIMC is used by control program functions such as the Internal Event Facility and the CIO Lost Interrupt Facility.

Note: This is the only state from which a processor switch should be attempted.

Utility (UTIL) State

UTIL state is defined as follows:

- Only Z action codes are accepted and only from valid CRAS terminals.
- The software clocks are stepping and have been adjusted to compensate for the time spent in 1052 state.
- Time-initiated entries are activated.
- The time available supervisor (TAS) is not active.
- GFS is not active. An attempt to issue a GETSC, GETLC, GETFC, or RELFC macro results in a system error.
- Keypoint update is active.
- DASD and tape lost interrupts are active.
- SEND and CRAS macros will be routed to the system console regardless of the line number, except for 3270 local output. Multisegment messages are not allowed and, if found, the remaining segments will be lost.
- The long message transmitter and output message transmitter may still have output messages in queue, but these queues are not being serviced. See *TPF Data Communications Services Reference* and *TPF ACF/SNA Data Communications Reference* for more information about these programs.
- This state can be entered only from 1052 state using the ZCYCL command.
- You can cycle only to 1052 state from UTIL state. You cannot cycle to any other state.
- Interrupts are accepted from 3270 local devices but messages will be rejected unless the terminal is logged on to the system message processor or it is a log in message to the system message processor. See *TPF Data Communications*

Services Reference for more information about the system message processor. The device must also be listed in the CRAS table for valid input. Multisegment (long) input messages and certain LOGU and LOGP functions will also be rejected because they use pool records. Incorrect input will result in the following message:

```
SYSTEM RESTRICTED, RETRY IN 5 MIN.
```

CRAS State

CRAS state is defined as follows:

- All terminals, whether they are a part of the TPF system or VTAM, are allowed to log on to the TPF system applications. However, non-CRAS authorized terminals have their application access restricted, and if you send a message to any application from a non-CRAS terminal, you will receive the following message:

  ```
  SYSTEM RESTRICTED, RETRY IN 5 MIN.
  ```
- The software clocks are stepping and have been adjusted to compensate for the time spent in 1052 state.
- Time-initiated entries are activated.
- The time available supervisor (TAS) is not active.
- GFS is active.
- Keypoint update is active.
- All active synchronous data link control (SDLC) lines and binary synchronous communication (BSC) lines are being polled. Synchronous link is active.

Message Switching State

Message switching state is similar to CRAS state with the following exceptions:

- All lines are active.
- For terminals associated with the reservation (RES) application, the only high-speed messages accepted in the system are messages with a primary action code of Y, Z, or O. If the primary action code is O, an additional check is performed so that only those messages associated with message switching are allowed. All other high-speed entries are rejected. All low-priority (type B) input is accepted. However, only message switching entries are processed. All low-priority (application) reservation messages are queued until the TPF system is cycled to NORM state.

Normal (NORM) State

NORM state is defined as follows:

- NORM state allows you to start all system and application functions. All entries are allowed.
- This is the only state in which the time available supervisor (TAS) is active.
- This is the only state in which catastrophic error recovery will be attempted, when active.

Fallback

If a performance loss or system failure occurs, it may be desirable to operate at a less efficient level rather than changing back to switchover procedures.

Device Fallback

You can perform manual fallback procedures for all hardware devices by issuing a command from a designated CRAS terminal. There are also some automatic fallback procedures. To fall back manually, you can change device assignments from any valid CRAS terminal. Automatic fallback is provided for the prime CRAS, RO CRAS, and tape devices. The switching, in this case, is automatically controlled by the system and you are informed of the action that is taken.

Control Unit Fallback

For DASD, you can perform a control unit fallback by using the module status change command program and moving each affected module to a different control unit.

Communication Device and Line Fallback

Manual and automatic procedures exist for communication device and line fallback. Manual fallback is performed by entering a command from a CRAS terminal. Automatic fallback for the system console is performed by monitoring online errors and device traffic. When excessive error conditions occur, an alternate CRAS device is assigned. There is no automatic fallback for communication devices and lines other than prime and RO CRAS. A defective device is automatically taken offline and you are notified. You can manually assign any alternate device using a command. If there are no unimpaired devices on a line, the line is taken offline automatically. A line stays active as long as there is one device running on it.

Manual Dumps

There are two ways to start a manual dump:

1. Enter the ZDUMP command
2. Perform a system restart from the CPU console.

Method 1 has the advantage of being noncatastrophic but has the disadvantage of using some main storage blocks that may contain information relevant to the problem. Therefore, this method is more commonly used in a test environment to take a snapshot dump after a test.

Method 2 is more commonly used in an operational environment whenever the system must be IPLed. Although it starts a catastrophic dump, a software IPL follows automatically, if possible, with the added advantage of a system reset (to clear the channels).

Method 2 is used if:

- The system failed.
- The system *hung* (was suspended).
- All lines stopped polling.

Mode Selection

The control program operates in either normal mode or test mode. The mode selection affects system files and procedures, and is governed by the mode option byte in keypoint record A. The mode options can be changed at load time by patching keypoint record A. Mode selections result in the following conditions:

	Normal Mode	Test Mode
Tapes	• The CLOSE macro causes a rewind and unload (except as noted in the TCLSC macro). • Tape position is not affected by a restart.	• All tapes that would normally be unloaded by a CLOSE or DISMOUNT macro are only rewound.
Control Program	• No main storage is allocated for test driver keypoints. • Driver restart is automatically bypassed. • ZTEST command entries result in an `INVALID ENTRY` response.	• Test drivers are started. • PTV is available for use.

System Error Options

The selective memory dump table (SMDT) and dump override tables control dump content. These tables determine which functional areas are included or excluded in a dump for a particular system error number.

For application errors it is possible to reduce the output to only produce a dump of the ECB virtual memory.

A duplicate error suppression option is available that prevents main storage dumps if the program name and version is identical to a previous error.

These and other self-explanatory options are specified in alter or display system error options and keypoint record A, and can be modified by patching at load time or by using commands.

The following commands control the production of the dump caused by a system error.

- Use the ZASER command to control:
  - Whether or not dumps are produced.
  - What device dumps are written to tape or a printer.
  - Whether notification of the error is written to the console or to tape (or both).
  - Whether or not data is presented to the dump data user exit.
  - Whether duplicate ECB dumps are suppressed.
  - The control program selection of applicable areas (versus snapshot dump user-specified main storage areas).
- Use the ZDSER command to display the system error options set by the ZASER command.
- Use the ZIDOT command to override the SMDT and existing override tables if you need more than the standard amount of data to debug a particular problem. Supply the system error number and the keywords that correspond to the areas of main storage that you want to include in the dump. For example, IMFST is a keyword that identifies the module file status table storage.
- Use the ZSTRC command to control the system trace options.
- Use the ZSPER command to change or display program event recording options online. Be careful when you use this command because it can affect system performance.

Control Program (CP) Error Messages

There are 5 types of error messages generated by the control program:
- Initial IPL or restart messages
- Hardware error messages
- System error messages
- The get file storage (GFS) facility error messages
- Lost interrupt messages.

These messages are for your information only. The action taken, for the most part, depends on local procedures and the nature of the error.

If a DASD or tape hardware error occurs, but is corrected by internal retry procedures, an error message notifies you of any possible hardware performance loss. If retry procedures are unsuccessful, a system error core dump will result with the hardware error message appended to the standard system error message. A dump is not taken if an irrecoverable channel check error occurs, but an error record with all pertinent data is written to the dump device. The system is then cycled to 1052 state (if it is not already in 1052 state).

See *Messages (System Error and Offline)* and *Messages (Online)* for more information about error messages.

Control Program Keypoint Records

The system configuration is specified in keypoint records. The keypoints are defined by DSECTs and are assembled. These records are used to initialize the system during restart and their file copy is updated whenever the core image counterparts are modified by the control program or by a command program.

Table 1. Control Program Keypoints

Keypoint	Macro Name	Function	Processor	SS	Initialized By	Residency	Demand Keypointable
Record A (CTKA)	CK1KE	Contains information required for system loading and for the initializer program.	Unique	Unique	SIP	File	No
Record B (CTKB)	CK9KC	Miscellaneous initialization and restart values, for example, clock status, VFA status, and DASD error thresholds.	Unique	Unique	SIP	Main storage	Yes
Record C (CTKC)	CK8KE	Status of Computer Room Agent Set (CRAS) attached terminals, initial Routing Control Application Table (RCAT) and Terminal Address Table (WGTA) control information.	Shared	Shared	SIP	Main storage	No
Record D (CTKD)	CK7KE	Status used by the synchronous link programs.	Unique	Shared	SIP	Main storage	Yes
Record E (CTKE)	CK6KE	Describes the non-SNA communications network.	Unique	Shared	SIP	File	No
Record I (CTKI)	IC0CK	Describes the status of all processors in a loosely coupled complex of the HPO feature.	Shared	Shared	SIP	File	No
Record M (CTKM)	MK0CK	Describes the status of each subsystem and each subsystem user.	Shared	Shared	SIP	Main storage	No
Record V (CTKV)	IDSCKV	Contains volume serial number ranges for the online modules, the copy module, and the loader general file.	Shared	Unique	SIP	File	No
Record 0 (CTK0)	CK0KE	Contains legal disk hardware addresses.	Shared	Shared	SIP	File	No
Record 1 (CTK1)	CK2KC	Contains the Tape Status Table (TSTB).	Unique	Shared	N/A	Main storage	Yes
Record 2 (CTK2)	CK2SN	Contains all the information in the system about the SNA configuration and the TCP/IP device parameters.	Unique	Shared	Source, contains no SIP provided inputs	Main storage	No
Record 3 (CTK3)	None	This keypoint is available for customer use.	Unique	Shared	Customer	File	No
Record 4 (CTK4)	VK4CK	This keypoint is available for customer use.	Shared	Unique	Customer	File	No
Record 5 (CTK5)	None	This keypoint is reserved for IBM use.	Shared	Shared	N/A	File	No

Table 1. Control Program Keypoints (continued)

Keypoint	Macro Name	Function	Processor	SS	Initialized By	Residency	Demand Keypointable
Record 6 (CTK6)	CJ6KP	Contains the DASD module status indicators.	Shared	Unique	SIP	File and main storage (see note 1)	No
Record 9 (CTK9)	CY1KR	Contains the status of the DASD pools.	Shared	Unique	Source, contains no SIP provided inputs	Main storage	No

Note:

1. The entire keypoint is file-resident; the first section of the keypoint is also main-storage-resident.
2. *Processor shared* means that there is one copy of the keypoint for all processors in a loosely coupled environment.
3. *Processor unique* means that there is one copy of the keypoint per processor in a loosely coupled environment.
4. *Subsystem (SS) shared* means that there is one copy of the keypoint residing in the BSS in an MDBF environment.
5. *Subsystem (SS) unique* means that there is one copy of the keypoint per subsystem in an MDBF environment.

User-Defined Commands

You can define your own commands for use with the TPF system using the ZFMSG command. For example, you can define commands that call or end user exits. You can also define your own command table (UMET) and command editor programs without impacting TPF commands.

The TPF system does not release commands that have a secondary action code of U; for example, ZU*xxx*. These commands are reserved for you, although you are not restricted to ZU*xxx*.

All the commands that you create are passed to the UME1 program segment. UME1 displays the following message when it receives control and causes the ECB to exit without performing any processing:

```
UME10000I NO USER COMMAND EDITOR AVAILABLE FOR ZUxxx
          OR ZDIAG COMMAND--ECB EXITED
```

For more information about defining your own commands and using the commands you define, see:

- "ZFMSG–Process Command Characteristics" on page 559
- *TPF Data Communications Services Reference*, for information about CRAS support and the system message processor.

Message Prefixing

There are times when, in a loosely coupled or a multiple database function (MDBF) environment, you will need to interrogate or update the status of processors and subsystems other than the one with which you are currently communicating. You may also have a need to communicate with a remote TPF processor to determine the status of some part of that system. The TPF system provides a prefixing facility to allow you to make the necessary inquiries and updates to other processors and subsystems to help you solve problems and be more productive. Figure 1 shows an example of a system comprised of a loosely coupled environment that includes processors B and C with MDBF, and uniprocessor A attached through a cross-domain link. Assume terminal 123B is logged on to SMPB, which is in the basic subsystem (BSS) of processor B.

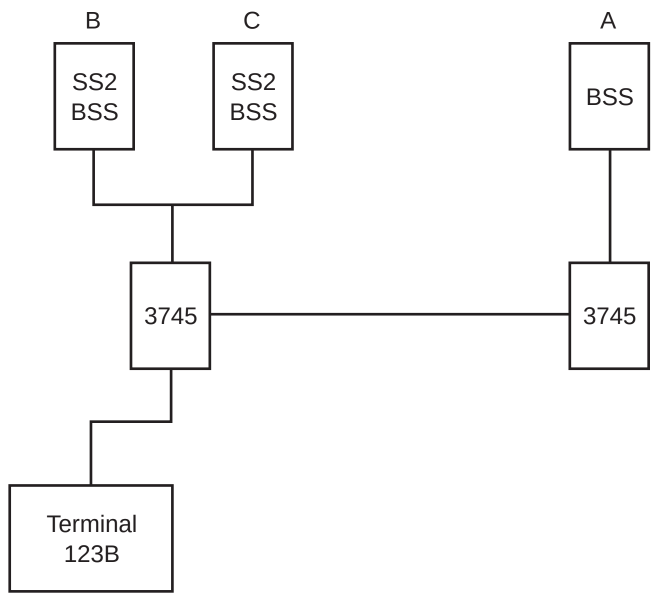


Figure 1. System Example

The following are the possible prefixes for the environment described in Figure 1:

Note: The *message* variable is the message that you want to enter.

- To retrieve status from processor C, enter:

  SMPC/*message*
- It is possible to send a message to any application (for example, AAAA) with the use of a prefix from a CRAS terminal logged on to SMP as long as the application can interpret the message, as follows:

  AAAA/*message*
- To send a message to any subsystem or subsystem user, enter:

  SS2/*message*
- To interrogate or update a subsystem on another processor, enter:

  SMPC/SS2/*message*
- In order to access the remote processor, enter:

  SMPA/*message*
- It is possible to send a message to all active processors in the loosely coupled complex by entering the prefix for the message as follows:

  ALL/*message*
- To send a message to a subsystem on all active processors in the loosely coupled complex, enter:

  ALL/SS2/*message*

Notes:

1. A prefixed command can be entered only from a CRAS terminal logged on to SMP.
2. In an MDBF environment, application prefixing is restricted to the basic subsystem.
3. To maintain compatibility with previous releases, a double slash (//) is accepted as the prefix delimiter.

4. You can enter the ALL prefix from the prime CRAS (PRC) or a local 3270 CRAS terminal only.

See “Prefixing Messages” on page 32 for more information about message prefixing.

Tape Support Commands

This section describes the TPF system tape support. It includes information about the following:
- General tape information
- Automatic tape mounting
- Tape library dataserver

General Tape Information

The TPF system addresses tapes by 3-character symbolic names. The ZTLBL command creates the 3-character symbolic name. The ZTMNT command assigns the tape name to a physical device and records the assignment in the tape status table (TSTB). These symbolic tape names are used by tape macros to access tapes.

All system tapes except ALT tapes must be standard label (SL) tapes. All output tapes are also standard label tapes. Unlabeled input tapes are nonlabel (NL) tapes. Use ZTLBL parameters LSL, LSU, or LNL to define the label attributes for a symbolic tape name. See “ZTLBL–Tape Label Maintenance” on page 1347 for more information.

The following identifies the symbolic tape names that are reserved by the TPF system. Applications can use all other 3-character alphanumeric labels for symbolic tape names.

Tape Name	**Description**
ADR	Recoup.
ALL	All tapes.
ALT	Alternate tapes for active real-time or general output tapes. When an active output tape that does not have a standby (backup) tape is at the end-of-volume, an automatic switch is made to an ALT tape. **Notes:** 1. ALT tapes do not need labels. 2. Mount as many ALT tapes as you want to the TPF system.
BKD	Recoup.
BX*x*	Capture and restore.
CDM	3705 dump.
CLD	3705 load.
DBF	Database reorganization (fixed records).
DBP	Database reorganization (pool records).
DBR	Database restore (PTV).
LKT	Synchronous link control (SLC) link trace.
MRT	3270 mapping support.
OLD	E-type program loader.

Tape Name	Description
PLM	LMT purge.
RCP	Recoup.
RPE	Recoup.
RPI	Recoup.
RTA	Primary real-time tape.
RTC	Data collection.
RTL	Real-time dump tape.
RTT–RTY	Logging/exception recording.
RT*x*	Any other real-time tape.
SDF	Pilot.
TLD	Auxiliary loader.
TUT	Test unit tape (PTV).
UTL	Unit record display.
XC*x*	Restore logging tape.
XXX	Module has already been assigned.
YYY	Available space in the tape status table.

Using Automatic Tape Mounting

Automatic tape mounting allows the TPF system to perform the following tasks without operator intervention:

- Mount alternate (ALT) tapes on write-enabled tape devices.
- Convert ALT tapes to active tapes for tape macro processing.
- Convert ALT tapes to standby tapes for tape switching. Automatically mounting ALT tapes improves tape switching during the dump process, which minimizes the number of tape devices used by a multiple volume tape dump.
- Identify unexpired and uninitialized write-enabled tapes as they are loaded into tape devices.

You determine the tapes that can be mounted on specific tape devices by defining tape groups, assigning tape labels (names) to the tape groups, and assigning tape devices to the tape groups. When a tape device is enabled for automatic tape mounting, the TPF system automatically mounts an ALT tape if an expired, write-enabled tape is loaded on the tape device. If the TPF system needs an active or standby tape, it searches for an eligible ALT tape and converts the tape.

To use automatic tape mounting, you must do the following:

- Define the appropriate tape groups
- Enable tape devices for automatic tape mounting.

After you define the tape groups, you can do the following:

- Enable, disable, or resume tape devices for automatic tape mounting
- Display or delete tape groups
- Display the tape devices and tape labels that are assigned to a tape group
- Assign or unassign tape devices and tape labels to and from tape groups.

Defining Tape Groups

Tape groups determine whether an ALT tape on a particular tape device can be used to satisfy a tape open or tape switch request for a particular tape label. By grouping tape labels and assigning tape devices to various tape groups, you can direct tape output without manually mounting active and standby tape pairs. You can also control the scope of the tape assignment to a single subsystem user or to all subsystem users.

One or more tape devices can be assigned to a tape group, and 1 or more tape labels can be assigned to a tape group. To use a tape group, you must do the following:

1. Define a tape group name by using the ZTGRP command.

   A tape group is specific to a processor. Therefore, you can use the same tape group name on different processors. The ALL and NONE tape groups are already defined by the TPF system and, therefore, cannot be specified as unique tape group names. Use the ALL tape group name to indicate that a tape device is assigned to all tape groups. Use the NONE tape group name to indicate that a tape label is not assigned to any tape group.
2. Assign one or more tape devices to the tape group by using the ZTDEV command.

   When you assign a tape device to a tape group, you choose the scope of the assignment:
   - Assign the tape device to the tape group for the current subsystem user or for all subsystem users.
   - Assign the tape device to all the tape groups for the current subsystem user or for all subsystem users.

   Each tape device can be assigned to a maximum of 31 tape groups, one of which can be the ALL tape group.

   If you place a tape device offline and then bring it back online (using the ZTVAR command), the tape group assignments for that tape device are retained.
3. Assign a tape label to the tape group by using the ZTLBL command.

   When you assign a tape label to a tape group, the assignment is made for the subsystem user in which the tape label is defined. The tape label is also associated with the tape devices assigned to that same tape group. In this case, the subsystem user assignment depends on how it was defined for the tape devices. For example, if the tape devices were assigned to the tape group for all subsystem users, the tape label is also associated with all the subsystem users when the tape is mounted on those tape devices.

   You can define a maximum of 254 tape groups for each processor. The same tape label can be assigned to different tape groups across different subsystem users. For example, you can assign the TAP tape label to the GENERAL tape group in the SSU1 subsystem user and also assign it to the XYZ tape group in the SSU2 subsystem user.

Enabling Automatic Tape Mounting for a Tape Device

A tape device can be:

- Assigned to a tape group, which determines the tape labels that can automatically use the tape device to satisfy a tape open or tape switch request.
- Enabled for automatic tape mounting, which determines whether or not ALT tapes are automatically mounted on the tape device.

- Both assigned and enabled, which allows tape switching and macro processing requests to be handled automatically.

To determine what you should do with your tape devices, consider whether or not you need to direct your output to specific tape devices.

- If you do not need to direct your output to specific tape devices, you do not need to assign tape labels or tape devices to tape groups. In this case, enabling automatic tape mounting for a tape device allows ALT tapes to be mounted automatically. This allows the tape device to be used for tape switching requests without requiring an operator to mount standby tapes.
- If you need to direct your output to specific tape devices, you need to assign tape labels and tape devices to tape groups. In this case, enabling automatic tape mounting for a tape device allows the tape device to be used for tape switching and macro processing requests.

After a tape device is enabled for automatic tape mounting, the TPF system can automatically mount a tape as an ALT tape when a tape is loaded on the tape device and the tape device is ready. The tape is not mounted as an ALT tape if any of the following conditions are true:

- Automatic tape mounting is suspended for the tape device, which indicates that a TPF system process that temporarily prevents automatic tape mounting is using that tape device.
- The tape is not write-enabled.
- The tape does not contain a standard VOL1 label.
- The header labels on the tape indicate that the tape is not expired.

When an ALT tape is automatically mounted, it is mounted in the format that the tape was last recorded.

If you enable automatic mounting for a tape device that is ready and has a tape loaded but not mounted, the same process is performed that would have been performed if the tape had been loaded and the tape device had been ready. The tape is then mounted as an ALT tape as long as none of the conditions that prevent automatic tape mounting are true.

Enabling, Disabling, and Resuming Automatic Tape Mounting for a Tape Device

Use the ZTDEV command to enable, disable, or resume automatic tape mounting for a tape device.

Displaying or Deleting Tape Groups

Use the ZTGRP command to display or delete tape groups.

You cannot delete a tape group if a tape label that is defined for an active subsystem user is assigned to that tape group.

If a tape device is assigned to a tape group when you delete it, the tape device is unassigned from that tape group.

During tape restart for a hardware initial program load (IPL) or a nonfast IPL, all tape labels for all active subsystem users are checked to ensure that they do not refer to any tape groups that were deleted while the subsystem user may have been inactive. Any tape label that is assigned to a deleted tape group is reassigned to the NONE tape group.

Displaying Tape Devices

Use the ZTDEV command to display the tape groups to which a tape device is assigned in the current subsystem user. The status of automatic tape mounting for a tape device is also displayed.

Assigning and Unassigning Tape Labels

Use the ZTLBL command to assign a tape label to a new tape group or to unassign a tape label from a tape group.

Planning Tape Group Definitions

Knowing the search priorities used by the TPF system during automatic tape mounting can help when you define your tape groups. The TPF system uses one set of priorities for tape switching and ALT tape conversion and another set of priorities for macro processing.

Tape Switching and ALT Tape Conversion Search Priorities: If there is no appropriate standby tape during a tape switch, the TPF system tries to locate an appropriate ALT tape by using the following search priorities:

- If the tape label is assigned to a tape group, the TPF system tries to locate an ALT tape that is mounted on a tape device that is:
  1. Assigned to the tape group in the same subsystem user in which the tape label is defined
  2. Assigned to the tape group for all subsystem users
  3. Assigned to all tape groups in the same subsystem user in which the tape label is defined
  4. Assigned to all tape groups for all subsystem users
  5. Not assigned to any tape group.
- If the tape label is not assigned to any tape group (that is, assigned to the NONE tape group), the TPF system tries to locate an ALT tape that is mounted on a tape device that is not assigned to any tape group.

If an appropriate ALT tape is found, it is converted to the appropriate standby tape and the tape switch continues. If no appropriate ALT tape is found, you are prompted to mount the appropriate standby tape.

Macro Processing Search Priorities: If a TOURC, TOUTC, TOPNC, or TSYNC macro is called and there is no appropriate active tape, the TPF system tries to locate an appropriate ALT tape by using the following search priorities:

- If the tape label is assigned to a tape group, the TPF system tries to locate an ALT tape that is mounted on a tape device that is:
  1. Assigned to the tape group in the same subsystem user in which the tape label is defined
  2. Assigned to the tape group for all subsystem users
  3. Assigned to all tape groups in the same subsystem user in which the tape label is defined
  4. Assigned to all tape groups for all subsystem users.
- If the tape label is not assigned to any tape group, the TPF system does not search for an ALT tape. This restriction gives you control over starting applications.

If an appropriate ALT tape is found, the tape is converted to the appropriate active tape and the macro is called again. If no appropriate ALT tape is found, you are prompted to mount the appropriate active tape. When you are prompted to mount a tape (that is, no appropriate ALT tape was found), you can do one of the following:

- Mount the appropriate active or standby tape, as requested by the TPF system.
- Perform some action to mount an ALT tape that satisfies the previous search criteria. This can include entering the ZTMNT command to mount an ALT tape or simply placing a tape on the appropriate automount-enabled tape device and making the tape device ready.

Using the Tape Library Dataserver

The *tape library dataserver* allows you to load and unload tapes without operator intervention. The *library manager* controls all automated library operations in the tape library dataserver. Many automated library operations, especially setup operations, must be performed at the library manager. See the appropriate tape library dataserver operator's guide for more information about these automated library operations.

Once the tape library dataserver is set up and a tape device in the TPF system is attached to the tape library dataserver, you can control automated library operations from the TPF system by using the ZTPLF commands.

When you enter the ZTPLF commands, you indicate which tape library dataserver you want to use by specifying the address of a tape device that is attached to the appropriate tape library dataserver. Any number of tape library dataservers can be set up for a host system. However, a tape device can be attached to only one tape library dataserver.

Tape Library Categories

It is important to understand the concept of *tape library categories* when discussing automated library operations. Tape library categories are groups of tapes that have common attributes; for example, tapes to eject, tapes that are new to the tape library dataserver, and tapes that are used to clean tape devices. These tape library categories identify how to use the tape and where to put it. For example, tapes that are added to the convenience-eject category have a status of eject pending. The tape library dataserver moves these tapes to a convenience output station.

The following information describes the tape library categories:

Category	Description	
X'0000'	Null	
X'0001'–X'FEFF'	General use	
X'FF00'–X'FFFF'	Reserved for automated library operations	
	X'FF00'	Insert
	X'FF10'	Convenience-eject
	X'FF11'	Bulk-eject (if the hardware is installed)
	X'FFF9'	Service-tape
	X'FFFA'	Manually-ejected
	X'FFFB'	Purge-tape
	X'FFFC'	Unexpected-tape
	X'FFFD'	Unknown-tape
	X'FFFE'	Cleaner-tape
	X'FFFF'	VOLSER-specific.

You can move tapes from one tape library category to another by using the ZTPLF MOVE command. Once a tape is added to a tape library category, it remains in that tape library category until you move it to a new tape library category, enter the ZTMNT command to mount the tape, or unload the tape. When the last two conditions occur, the tape is automatically removed from its current tape library category and added to the default tape library category.

Without tape library categories, you need to specify the name of the tape that you want to load. By using tape library categories, you can simply specify the *type* of tape you want to load; for example, a scratch tape. In this case, the tape library dataserver loads the first available scratch tape.

Different host systems can share the same tape library categories. For example, the TPF system can share tape library categories with an MVS system.

Adding a Tape to the Tape Library Dataserver

To add a new tape to the tape library dataserver, put the tape in the input station. The library manager adds the tape to the X'FF00' (insert) tape library category.

Moving Tapes from One Tape Library Category to Another

To remove a tape from one tape library category and add it to a different tape library category, enter the ZTPLF MOVE command.

For example, to move the AL1234 tape from the X'FF00' (insert) tape library category to the X'FFFF' (VOLSER-specific) tape library category, enter:

ZTPLF MOVE DEV-041C VSN-AL1234 FROM-FF00 TO-FFFF

Note: The tape device that is specified in the ZTPLF commands merely indicates which tape library dataserver to use. Remember that a tape device can be attached to only one tape library dataserver.

You can move a tape from one tape library category to another at any time. For example, if the purpose of a tape changes, it may be necessary to add the tape to a different tape library category.

Removing a Tape from a Tape Library Category

A tape is automatically removed from its current tape library category and added to the default tape library category when you mount the tape by entering the ZTMNT command or when you unload the tape.

Reserving a Tape Library Category

You can define your own tape library categories. For example, you can define a tape library category for scratch tapes. To do this, enter the ZTPLF RESERVE command to reserve a general use tape library category (X'0001'–X'FEFF'). You can use this tape library category for scratch tapes.

For example, to reserve a general use tape library category in the tape library dataserver that is attached to the 041C tape device, enter:

ZTPLF RESERVE DEV-041C

A message is displayed to indicate which tape library category was reserved. You can enter the ZTPLF MOVE command to add tapes to this tape library category. Other host systems can use a tape library category that you reserved. For example, other host systems can add tapes to a reserved tape library category, load the tapes in a reserved tape library category, and unload the tapes in a reserved tape library category.

Use the ZTPLF RELEASE command to return the reserved tape library category to general use when you are finished with it.

Loading Tapes Using the Tape Library Dataserver

If a job requires a particular type of tape, you can use the tape library dataserver to add these tapes to the appropriate tape library category and then automatically load the necessary tapes.

To automatically load only one tape from a tape library category, enter the ZTPLF LOAD command.

For example, to automatically load the first available scratch tape from the tape library dataserver that is attached to the 041C tape device, enter:

ZTPLF LOAD DEV-041C CAT-1A00

Note: This example assumes that scratch tapes were added to the X'1A00' tape library category.

You can also continuously load ***all*** the tapes in a tape library category one after the other by entering the ZTPLF FILL command.

For example, to continuously load all the scratch tapes one after the other to the 041C tape device, enter:

ZTPLF FILL DEV-041C CAT-1A00

After the first tape in the tape library category is rewound and unloaded, the tape library dataserver automatically loads the next tape in the tape library category, and so on, until there are no more tapes in the 1A00 tape library category.

Note: Remember that a tape is automatically removed from its current tape category when you mount or unload that tape. Therefore, if a tape in the 1A00 tape library category is mounted on or unloaded from another tape device, it is automatically removed from the 1A00 tape library category at that time and will not be loaded on the 041C tape device. This strategy allows you to continuously load a tape library category using one or more tape devices.

If the tape library dataserver is continuously loading the tapes in a tape library category, it preloads the tapes to the automatic cartridge loader (ACL) if the ACL is installed and a tape is currently loaded in the tape device. Otherwise, the tape library dataserver loads the tapes directly to the tape device when the tape device is available.

Unloading Tapes Using the Tape Library Dataserver

If you decide that you no longer want to use the tape that is loaded in a tape device or the tape library category that is being continuously loaded to a tape device, you can enter the ZTPLF UNLOAD command to automatically unload the tapes. For example, to automatically unload a single tape from the 041C tape device, enter:

ZTPLF UNLOAD DEV-041C

If, in this case, the entire tape library category was being continuously loaded to the 041C tape device, the tape library dataserver unloads the current tape from the tape device and then automatically loads the next tape in the tape library category.

As another example, to automatically unload the tape that is currently loaded in the 041C tape device and also cancel the continuous load of the other tapes in that tape library category, enter:

ZTPLF UNLOAD DEV-041C ALL

If the tape that is currently loaded in a tape device is mounted, enter the ZTOFF command to dismount the tape before you unload it.

Note: If you try to unload a tape library category without first dismounting the tape that is currently loaded in the tape device, the tape library dataserver cancels the continuous load of the tapes in that tape library category but ***does not*** unload the tape that is currently loaded in the tape device.

Querying the Status of the Tape Library Dataserver

To query the status of the tape library dataserver, enter the ZTPLF QUERY command. You can display the following information:

- Status of a tape device
- Status of a tape
- List of tapes that were added to a specific tape library category
- List of tape library categories that were reserved.

For example, to display the status of the AL1234 tape, including the name of the tape library category to which that tape was added, enter:

ZTPLF QUERY DEV-041C VSN-AL1234

As another example, to display a list of the tapes that were added to the X'1A00' tape library category, enter:

ZTPLF QUERY DEV-041C CAT-1A00

The tapes are displayed in the order that they were added to the tape library category. You can also display the list alphabetically or begin the list after a certain number of tapes.

Detecting Lost Interrupts and Stalled Tape Module Queues

When the TPF system detects a long or lost interrupt, it stops waiting and performs a tape switch or reports a permanent error to the application.

When the TPF system detects a stalled tape module queue with no pending interrupt, it performs a tape switch or reports a permanent error to the application. You can adjust the time-out value for the stalled tape module queue by using the ZCTKA command.

The default for the time-out value is 0, which indicates that the system does not check for a stalled module queue condition. Change this value only if you experience repeated system failures because of out-of-storage conditions and when dumps indicate that module queues for one or more tape devices are not active. Report this condition to your IBM representative.

Additional Considerations

When operating in a multiple database function (MDBF) or loosely coupled processor environment, consider the following:

- The processor resource ownership table (PROT) must be maintained for tape devices that belong to a processor. This task is automatically handled by tape restart and by the ZTVAR command.
- Some commands can affect all subsystem users and therefore can be entered only from the basic subsystem (BSS). For example, all commands that handle real-time and ALT tapes and any command that use the ALL parameter are restricted to the BSS. These commands include:
  - ZTGRP with the DEFINE parameter specified
  - ZTGRP with the DELETE parameter specified
  - ZTICL
  - ZTOCU
  - ZTOFF ALL
  - ZTVAR
  - ZTWTM.

TPF Commands

This chapter contains an alphabetic listing of the TPF commands. The description of each command contains the following information:

Purpose: Describes how and why you use the command.

Requirements and Restrictions: Lists the tasks that you must perform before you can enter the command and any restrictions for the command.

Format: Provides a syntax (or railroad track) diagram for the command and a description of each parameter and variable. See “How to Read the Syntax Diagrams” on page xxv for more information about syntax diagrams.

Additional Information: Lists miscellaneous information about the command.

Examples: Provides one or more examples that show you how to use the command.

Related Information: Lists other books and commands that contain related information.

LOGx–Log On or Log Off a Display Device to an Application

Use this command to:

- Logically connect or disconnect a non-SNA display device to a specified application program
- Display or delete unsolicited messages sent that are to a non-SNA display device.

Requirements and Restrictions

None.

Format

```
▶▶──┬─LOGI── ──appl──────┬──────────────────────────────────────────▶◀
    ├─LOGO───────────────┤
    ├─LOGU── ──┬─appl─┬──┤
    │          └─MENU─┘  │
    └─LOGP── ──appl──────┘
```

LOGI
connects the non-SNA display device to the specified application program.

LOGO
disconnects the non-SNA display device from an application program.

LOGU
displays the unsolicited messages sent from a CRAS console.

LOGP
deletes unsolicited messages from the unsolicited message queues.

appl
is a 1- to 4-character application name.

MENU
displays information about the pending unsolicited messages.

Additional Information

None.

Examples

Information about the pending unsolicited messages is displayed in the following example, where:

APPL
is the name of the application that sent the unsolicited messages.

CTR
is the number of unsolicited messages sent by the application.

```
User:   LOGU MENU

System: UMSG(S) PENDING AS FOLLOWS:
        APPL CTR    APPL CTR    APPL CTR
        SMPB 003
```

The first unsolicited message that is sent by the SMPB application is displayed in the following example.

```
User:   LOGU

System: THIS IS THE FIRST MESSAGE FROM SMPB.
```

Related Information

See *TPF Data Communications Services Reference* for more information about the log processor for non-SNA display devices and about unsolicited messages.

Prefixing Messages

Use message prefixes to direct messages from a computer room agent set (CRAS) console to a specific application, subsystem, or subsystem user, or to all active processors in a loosely coupled complex.

Requirements and Restrictions

The CRAS console must be logged on to the system message processor (SMP).

Format

```
      ┌─/──────────────┐
      ▼                │
►►──┬─┬─appl──────┬────┴─┬──/message──────────────────────►◄
    │ ├─subsystem─┤      │
    │ └─ssuser────┘      │
    └─ALL────────────────┘
```

appl
: sends the message to the specified application program, where *appl* is a 1- to 4-character alphanumeric application name.

subsystem
: sends the message to the specified subsystem where *subsystem* is a 1- to 4-character alphanumeric subsystem name.

ssuser
: sends the message to the specified subsystem user where *ssuser* is a 1- to 4-character alphanumeric subsystem user name.

ALL
: sends the message to all the active processors in the loosely coupled complex.

message
: is the message that you want to send.

Additional Information

- When you direct a message to a subsystem or a subsystem user, the message is actually sent to the CVA1 program in the subsystem or subsystem user for processing.
- To maintain compatibility with previous TPF system releases, a double slash (//) is also accepted as the prefix delimiter.

Examples

In the following example, the ZDSYS command is entered for all the active processors in the loosely coupled complex.

```
User:   ALL/ZDSYS

System: CSMP0097I 19.29.48 CPU-B SS-BSS  SSU-HPN  IS-01
        DSYS0001I 19.29.48 THE SYSTEM IS IN 1052 STATE
        CSMP0097I 15.16.45 CPU-C SS-BSS  SSU-HPN  IS-01
        DSYS0001I 15.16.45 THE SYSTEM IS IN 1052 STATE
```

In the following example, the ZRIPL command is entered for the processor where the SMPC application is running.

```
User:   SMPC/ZRIPL

System: CSMP0097I 09.03.21 CPU-C SS-BSS  SSU-HPN  IS-01
        RIPL0000I 09.03.21 ZRIPL-OK
```

Related Information

See “Message Prefixing” on page 16 for more information about prefixing commands.

Z Mxyy–Modify Scroll Display

Use this command to scroll the information displayed on a display device.

Requirements and Restrictions

None.

Format

```
►►──Z── ──M──┬─┬─D─┬──┬───────┬─┬──────────────────────────────►◄
             │ └─U─┘  └─lines─┘ │
             ├─T────────────────┤
             └─B────────────────┘
```

D scrolls the information down the specified number of lines.

U scrolls the information up the specified number of lines.

lines
is the number of lines from 0–99. If you do not specify the number of lines, the information scrolls a full page.

T scrolls to the top of the information.

B scrolls to the bottom of the information.

Additional Information

None.

Examples

In the following example, information in the display is scrolled down 3 lines. Notice the vertical bar (|) that marks the end of each display to indicate that there is more information.

```
User:   ZDCOR 1000.400

System: DCOR0010I 09.13.45 BEGIN DISPLAY
         00001000- 900792A8 58D00324 58F0030C 4AF0E000 ........ .0...0..
         00001010- 58A0F000 0B0A5860 B1805860 624C483E ..0....- ...-....
         00001020- 00001B22 4140000C 1D248930 00061A63 ..... .. ........
         00001030- 4130B040 5630B03C 0B030000 80000000 ... .... ........
         00001040- 982492B0 50ED0054 50FD0050 58F0BB68 ........ .....0..
         00001050- 58F0F0F8 0DEF58ED 00545869 02C00B0A .008.... .....|

User:   Z MD3

System:  00001030- 4130B040 5630B03C 0B030000 80000000 ... .... ........
         00001040- 982492B0 50ED0054 50FD0050 58F0BB68 ........ .....0..
         00001050- 58F0F0F8 0DEF58ED 00545869 02C00B0A .008.... ........
         00001060- 00001014 00000002 00001070 0000105E ........ ........
         00001070- 070058F0 B07807FF 000DFD34 47F0B072 ...0.... .....0..
         00001080- 58109410 48101052 482F0008 17121412 ........ .....|
```

Related Information

See *TPF Data Communications Services Reference* for more information about the system message processor and the modify scroll display.

Z3705 CLD–Load 3705 Modules to the TPF System

Use this command to load the 3705 load modules that are required to load, dump, and operate a 3705 control unit. These load modules are loaded from a load file (CLD tape) that is created offline.

Requirements and Restrictions

None.

Format

```
►►──Z3705 CLD──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The 3705 load modules are loaded to the TPF system in the following example.

```
User:   Z3705 CLD

System: 37050029I 06.56.56 PROGRAM CXZNPH1 LOADED
        37050029I 06.56.56 PROGRAM CXZNPH2 LOADED
        37050029I 06.56.56 PROGRAM IFLLD1P2 LOADED
        37050029I 06.56.56 PROGRAM CXWMINI1 LOADED
        37050029I 06.56.56 PROGRAM EPGENA LOADED
        37050029I 06.56.56 PROGRAM EPGENB LOADED
        37050029I 06.56.56 PROGRAM EPGENC LOADED
        37050029I 06.56.56 PROGRAM IFL3705A LOADED
        37050029I 06.56.56 PROGRAM IFL3705D LOADED
        37050029I 06.56.56 PROGRAM IFL3705E LOADED
          3705    CA      -LOAD MODULE-    STATUS
          ADDR   TYPE     NAME     TYPE      IND.
        ------------------------------------------
          01C     4               -        31
          01A     4               -        31

          THE FOLLOWING LOAD MODULES ARE ON FILE
              EPGENA     -EP
              EPGENB     -EP
              EPGENC     -EP
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

Z3705 DMP–Dump 3705 Control Unit

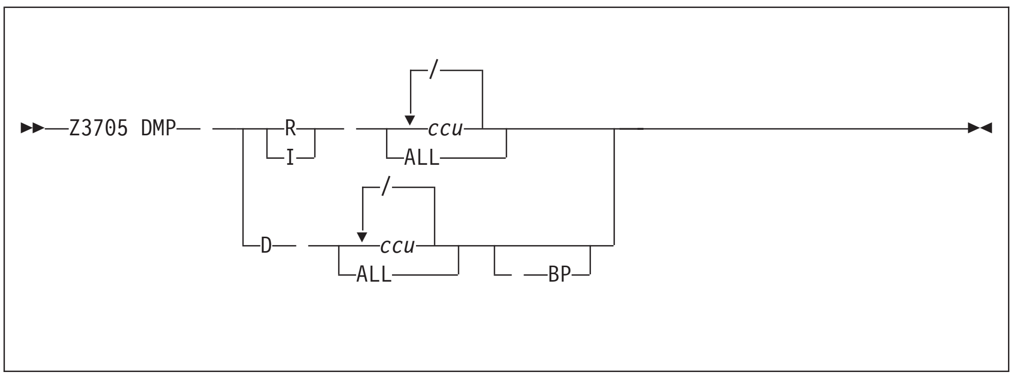
Use this command to perform a dump for a 3705 control unit. If the TPF system is above CRAS state, the dump is written to file storage. If the TPF system is below CRAS state, that is, the get file storage (GFS) facility is not active, the dump is written to a CDM tape.

You can also use this command to turn on or turn off the automatic dump option. The *automatic dump option* performs a dump when an initial program load (IPL) is automatically performed on a 3705 control unit. Enter the Z3705 IPL command to turn on or turn off the automatic IPL option.

Requirements and Restrictions

- If the TPF system is below CRAS state, mount a CDM tape before you enter this command.
- Enter the Z3705 IPL command to perform an IPL for the 3705 control unit after a dump is performed.

Format



R turns on the automatic dump option for a 3705 control unit.

I turns off the automatic dump option for a 3705 control unit.

D performs a dump for a 3705 control unit.

ccu
is a 3-digit hexadecimal channel control unit address. For example, you can specify the native subchannel (NSC) address of the control unit.

ALL
specifies all 3705 control units.

BP
performs a dump for a 3705 control unit when the system is in NORM state and the control unit is online and in emulation program (EP) mode.

Additional Information

None.

Examples

The automatic dump option is turned on for the specified 3705 control unit in the following example.

```
User:   Z3705 DMP R 01C

System: 37050023I 06.56.56 C.U. 01C AUTO-DUMP RESTORED OK
```

A dump is performed for all of the 3705 control units in the following example.

```
User:   Z3705 DMP D ALL

System: 37050035I 15.14.46 C.U. 01C DUMPED OK, DUMP NUMBER- 0002
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

Z3705 DSP–Display 3705 Status

Use this command to display information about the 3705 load modules.

Requirements and Restrictions

Before you enter this command, you must enter **Z3705 RNT ALL**.

Format

```
►►──Z3705 DSP──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

Information about the 3705 load modules is displayed in the following example, where:

3705 ADDR
: is the channel control unit address.

CA TYPE
: is the channel adapter type.

LOAD MODULE NAME
: is the module name.

LOAD MODULE TYPE
: is the load module type.

STATUS IND.
: are the status indicators from the 3705 keypoint (CCKP), where:

Bit	Value	Description
0	0	Automatic dump option is turned on.
0	1	Automatic dump option is turned off.
1	0	Load/dump is not in progress.
1	1	Load/dump is in progress.
2–3	00	Type 1 channel adapter
2–3	11	Type 4 channel adapter
4–7	0000	No control unit restrictions
4–7	0001	EP-only control unit.

```
User:   Z3705 DSP

System: 3705    CA      -LOAD MODULE-    STATUS
        ADDR   TYPE     NAME     TYPE      IND.
        ________________________________________
        01C     4     EPGEN3     -EP      B1
        01A     4                -        B1
        0BC     4                -        B1
        0BD     4                -        B1
        715     4                -        B1
        716     4                -        B1
        711     4                -        B1
        712     4                -        B1

        THE FOLLOWING LOAD MODULES ARE ON FILE
            EPGEN1     -EP
            EPGEN2     -EP
            EPGEN3     -EP
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

Z3705 IPL–IPL 3705 Control Unit

Use this command to perform an initial program load (IPL) on a 3705 control unit. You can also use this command to:

- Associate a load module with a 3705 control unit without performing an IPL.
- Run diagnostics on a 3705 control unit.
- Turn on or turn off the automatic IPL option for a 3705 control unit. The *automatic IPL option* automatically performs an IPL on a 3705 control unit when a critical error occurs. Enter the Z3705 DMP command to specify whether or not a dump should also be performed.

Requirements and Restrictions

None.

Format

```
                           ┌─/───┐
                           ▼     │
►►──Z3705 IPL── ──┬─┬─I─┬─ ──ccu─┴──┬─────────┬─┬──────────────────────────►◄
                  │ └─R─┘           └─ ──NL─┘ │
                  │ ┌─/──────────┐            │
                  │ ▼            │            │
                  └─ccu-module───┴─┬───────┬──┘
                                   ├─ ──BP─┤
                                   ├─ ──NL─┤
                                   ├─ ──DG─┤
                                   └─ ──DB─┘
```

I turns on the automatic IPL option for a 3705 control unit.

R turns off the automatic IPL option for a 3705 control unit.

ccu
is a 3-digit hexadecimal channel control unit address. For example, you can specify the native subchannel (NSC) address of the control unit.

module
is a 1- to 8-character load module name.

BP
forces an IPL on a 3705 control unit when the control unit is online and the TPF system is in CRAS state or higher.

NL
associates a load module with a 3705 control unit without performing an IPL.

Note: If you specify the NL parameter for an automatic IPL, the previous load module is associated with the specified 3705 control unit without physically performing an IPL. Any existing sessions are cleaned up.

DG
performs diagnostics on a 3705 control unit by loading and running the initial test program before loading the load modules. This takes approximately 30 seconds.

DB
forces diagnostics to be performed on a 3705 control unit.

Additional Information

None.

Examples

An IPL is performed on the specified 3705 control unit in the following example.

```
User:   Z3705 IPL 01C-EPGENA

System: 37050045I 06.56.56 IPL COMPLETE- EPGENA NOW LOADED IN C.U. 01C
```

In the following example, the EPGENA load module is associated with the specified 3705 control unit without performing an IPL.

```
User:   Z3705 IPL 01C-EPGENA NL

System: 37050049I 06.56.56 EPGENA ASSOCIATED WITH C.U. 01C
```

The automatic IPL option is turned on for the specified 3705 control unit in the following example.

```
User:   Z3705 IPL I 01C

System: 37050114I 06.56.56 C.U. 01C AUTO-IPL INHIBITED OK
```

Related Information

- See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.
- See *3704/3705 Control Program Generation Utilities Guide* for more information about performing diagnostics on a control unit.

Z3705 PRG–Purge 3705 Dump from Online System

Use this command to remove all 3705 dumps from file storage on the TPF system and write them to a dump tape (CDM) file.

Requirements and Restrictions

None.

Format

```
►►──Z3705 PRG──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The 3705 dumps in file storage are moved to a dump tape in the following example.

```
User:   Z3705 PRG

System: 37050055I 15.14.53 COMPLETE - CDM TAPE DETAILS TO RO
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

Z3705 RNT–Reinitialize 3705 Status Indicators

Use this command to initialize a 3705 control unit. The 3705 load/dump in progress indicator for the control unit in the 3705 keypoint (CCKP) is turned off and the I/O queues are cleared.

Requirements and Restrictions

Enter this command each time you load the 3705 keypoint.

Format

```
                       ┌─/───┐
                       ▼     │
►►──Z3705 RNT──  ──┬──ccu──┴─┬──────────────────────────────────────►◄
                   └─ALL─────┘
```

ccu
: is a 3-digit hexadecimal channel control unit address. For example, you can specify the native subchannel (NSC) address of the control unit.

ALL
: initializes all the 3705 control units.

Note: The program version of the 3705 keypoint is copied to a fixed file record.

Additional Information

None.

Examples

All of the 3705 control units are initialized in the following example.

```
User:   Z3705 RNT ALL

System: 37050053I 11.31.50 C.U. 01A - STATUS REINITIALIZED
        37050053I 11.31.50 C.U. 01B - STATUS REINITIALIZED
        37050053I 11.31.50 C.U. 01C - STATUS REINITIALIZED
```

Two 3705 control units are initialized in the following example.

```
User:   Z3705 RNT 01D/01E

System: 37050053I 11.31.50 C.U. 01D - STATUS REINITIALIZED
        37050053I 11.31.50 C.U. 01E - STATUS REINITIALIZED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

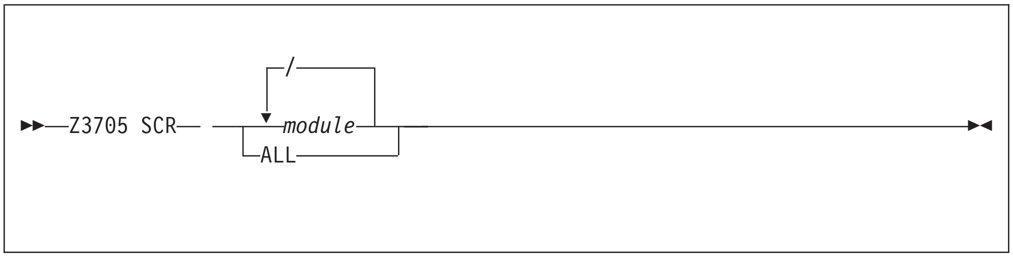
Z3705 SCR–Scratch 3705 Module References

Use this command to remove the references to one or more load modules from the 3705 keypoint (CCKP).

Requirements and Restrictions

None.

Format



module
: is a 1- to 8-character load module name.

ALL
: removes the references to all the load modules from the 3705 keypoint.

Additional Information

None.

Examples

In the following example, the references to all the load modules are removed from the 3705 keypoint.

```
User:   Z3705 SCR ALL

System: 37050034I 06.56.56 ALL LOADIDS SCRATCHED OK
```

Related Information

See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.

Z3705 TRC–3705 EP Dynamic Line Trace

Use this command to run the emulation program (EP) line trace function of the Dynamic Dump utility. You can:

- Activate the EP line trace function
- Start or stop different line traces for different subchannel addresses
- Send the trace data to the TPF system and then write it to a DYN tape
- Stop sending trace data to the TPF system
- Stop the EP line trace function.

Requirements and Restrictions

- The TPF system must be in CRAS state or higher.
- The EP must be loaded on the 3705 control unit that you want to trace.
- The 3705 load and dump functions cannot be active.
- You must start the EP line trace function before you can start any line traces. Once you start the EP line trace function, you have 60 seconds to start a line trace. If you do not start a line trace in 60 seconds, the EP trace function ends.
- Mount a DYN tape before you start sending trace data to the TPF system.
- Once you start sending trace data to the TPF system, you cannot start another line trace until you stop sending the trace data to the TPF system.
- Once you stop sending trace data to the TPF system, you have 60 seconds to start another line trace or the EP trace function ends.
- You can trace a maximum of 16 subchannels at one time.

Format

```
►►──Z3705 TRC── ──┬─NSC── ──ccu──────────────┬──────────────────────────────►◄
                  ├─OPT── ──┬─10subaddr─┬──┤
                  │         ├─11subaddr─┤  │
                  │         ├─20subaddr─┤  │
                  │         ├─21subaddr─┤  │
                  │         ├─30subaddr─┤  │
                  │         ├─31subaddr─┤  │
                  │         ├─7000──────┤  │
                  │         ├─7100──────┤  │
                  │         ├─70FF──────┤  │
                  │         └─71FF──────┘  │
                  ├─DYN────────────────────┤
                  ├─STP────────────────────┤
                  └─END────────────────────┘
```

NSC *ccu*
: starts the EP line trace function on the 3705 control unit, where *ccu* is the 3-digit hexadecimal channel address of the 3705 control unit.

OPT
: starts or stops a line trace for the specified subchannel address, where:

10*subaddr*
starts a level 2 line trace for the specified 2-digit hexadecimal subchannel address.

11*subaddr*
stops the level 2 line trace for the specified 2-digit hexadecimal subchannel address.

20*subaddr*
starts a level 3 line trace for the specified 2-digit hexadecimal subchannel address.

21*subaddr*
stops the level 3 line trace for the specified 2-digit hexadecimal subchannel address.

30*subaddr*
starts a level 2 and level 3 line trace for the specified 2-digit hexadecimal subchannel address.

31*subaddr*
stops the level 2 and level 3 line trace for the specified 2-digit hexadecimal subchannel address.

7000
starts a level 3 line trace for the trace-defined subchannels.

7100
stops the level 3 line trace for the trace-defined subchannels.

70FF
starts a level 3 line trace for all subchannels.

71FF
stops the level 3 line trace for all subchannels.

DYN
sends the trace data to the TPF system and then writes the trace data to a DYN tape.

Note: Mount a DYN tape before you specify this parameter.

STP
stops sending trace data to the TPF system.

END
ends the EP line trace function and stops sending trace data to the TPF system.

Additional Information

- Level 2 interrupts are line data interrupts.
- Level 3 interrupts are time-out complete or channel data interrupts (such as initial selection data) or status.
- Level 1 error log entries are traced continuously after a level 3 trace is started.

Examples

The trace function is started for the specified control unit in the following example.

```
User:   Z3705 TRC NSC 01C

System: 37050103I 08.32.28 60 SEC.LIMIT ACTIVE
        37050098I 08.32.28 ENT NEXT OPT
```

A level 2 trace is started on subchannel 12 in the following example.

```
User:   Z3705 TRC OPT 1012

System: 37050098I 08.32.38 ENT NEXT OPT
```

A level 3 trace is started on all subchannels in the following example.

```
User:   Z3705 TRC OPT 70FF

System: 37050098I 08.42.53 ENT NEXT OPT
```

The trace data is sent to the TPF system and then written to a DYN tape in the following example.

```
User:   Z3705 TRC DYN

System: 37050101I 08.34.09 TRC BLOCK 0001 WRITTEN
        37050101I 08.34.10 TRC BLOCK 0256 WRITTEN
```

The trace data is no longer sent to the TPF system in the following example.

```
User:   Z3705 TRC STP

System: 37050100I 08.43.41 STOP ACKNOWLEDGED
        37050103I 08.43.41 60 SEC.LIMIT ACTIVE
        37050098I 08.43.41 ENT NEXT OPT
```

The trace function is ended in the following example.

```
User:   Z3705 TRC END

System: 37050102I 08.47.49 TRC ENDED
```

Related Information

- See *TPF Data Communications Services Reference* for more information about the 3705 communications controller.
- See *3704/3705 Control Program Generation Utilities Guide* for more information about the Dynamic Dump utility.

ZACLV–Alter CPU Loop and Create Macro Control Levels

Use this command to change the control levels for the CPU loop, the CREATE macros, or the input list. The CPU loop levels determine when to service the various lists in the loop program. The CREATE macro levels determine when to activate a CREATE macro. Main storage availability determines the levels in each case. You can use this command to tune or change the activity of the real-time system in a dynamic environment.

Requirements and Restrictions

- You can enter this command only for the basic subsystem (BSS).
- You can enter this command only from the prime CRAS.
- Only system programmers should use this command.

Format

```
                    ┌─  ───(1)──────┐
                    ▼               │
►►──ZACLV──  ───────┬─MXECBS─┬──────┴──-value──────────────────────────►◄
                    ├─MNFRAI─┤
                    ├─MNCMBI─┤
                    ├─MNECBI─┤
                    ├─MNSWBI─┤
                    ├─MNIOBI─┤
                    ├─MXECBD─┤
                    ├─MNFRAT─┤
                    ├─MNCMBT─┤
                    ├─MNECBT─┤
                    ├─MNSWBM─┤
                    ├─MNSWBD─┤
                    └─MNSWBX─┘
```

Notes:

1 You can also separate the parameters with a comma (,).

MXECBS
changes the maximum number of entry control blocks (ECBs) allowed in the TPF system.

MNFRAI
changes the minimum number of frames that must be available to service the input list.

MNCMBI
changes the minimum number of common blocks that must be available to service the input list.

MNECBI
changes the minimum number of ECBs that must be available to service the input list.

MNSWBI
changes the minimum number of system work blocks (SWBs) that must be available to service the input list.

MNIOBI
changes the minimum number of I/O blocks that must be available to service the input list.

MXECBD
changes the maximum number of ECBs in the TPF system that cannot be exceeded to service the deferred list.

MNFRAT
changes the minimum number of frames that must be available to activate the time available supervisor.

MNCMBT
changes the minimum number of common blocks that must be available to activate the time available supervisor.

MNECBT
changes the minimum number of ECBs that must be available to activate the time available supervisor.

MNSWBM
changes the minimum number of SWBs that must be available to activate a CREMC macro.

MNSWBD
changes the minimum number of SWBs that must be available to activate a CREDC macro.

MNSWBX
changes the minimum number of SWBs that must be available to activate a CREXC macro.

value
is a decimal number from 1–32 767.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZACLV HELP
  ZACLV ?
- Enter the ZDCLV command to display the current control levels for the CPU loop, the CREATE macros, and the input list.

Examples

New values are assigned to the MXECBS and MNFRAI parameters in the following example.

```
User:   ZACLV MXECBS-200 MNFRAI-100

System: ACLV0001I 15.28.54  STORAGE CONTROLS

             DESCRIPTION     VALUE   ZACLV PARM

          SYST   MAX ECB       200     MXECBS
          INPUT  MIN FRAME     100     MNFRAI
                 MIN COMMON     18     MNCMBI
                 MIN ECB       100     MNECBI
                 MIN SWB        38     MNSWBI
                 MIN IOB        51     MNIOBI
          DEFER  MAX ECB        40     MXECBD
          TAS    MIN FRAME     300     MNFRAT
                 MIN COMMON     59     MNCMBT
                 MIN ECB       200     MNECBT
          CREM   MIN SWB        26     MNSWBM
          CRED   MIN SWB        64     MNSWBD
          CREX   MIN SWB        64     MNSWBX
        NOTE: ABOVE DISPLAY REFLECTS UPDATED LEVELS
```

Related Information

- See *TPF Main Supervisor Reference* and *TPF Concepts and Structures* for more information about the CPU loop and the input list.
- See *TPF Main Supervisor Reference* and *TPF General Macros* for more information about the CREATE macros.

ZACOR–Alter Core

Use this command to change as many as 16 bytes of system virtual memory (SVM), also known as core memory.

Requirements and Restrictions

You must specify the IS parameter in a tightly coupled system when prefixing applies; for example, when you change storage in page 0 of an I-stream, or when you change the I-stream unique global areas.

Format

```
                                                         ┌─ ──Is-1──┐
►►──ZACOR── ──address── ──newdata──┬───────────────────┬─┼──────────┼──────►◄
                                   └─ ──Valdata-olddata─┘ └─ ──Is-xx─┘
```

address
: is a 1- to 8-digit (4-byte) hexadecimal SVM address.

newdata
: is the new data that replaces the old data. The new data must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes).

Valdata-*olddata*
: verifies that *olddata* matches the data being changed. If there is a discrepancy, no data is changed. The *olddata* variable must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes). The length of *olddata* can be different from the length of *newdata*.

  Note: If you do not specify this parameter, the data is changed without any verification.

Is-*xx*
: specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZACOR HELP
  ZACOR ?
- The TPF system displays the contents of storage before and after you change it.
- The changed area can be in the system heap but it is changed only if it is currently in use.
- Enter the ZDCOR command to display SVM.

Examples

In the following example, 4 bytes of data in the prefix area of I-stream 1 are changed.

ZACOR

```
User:   ZACOR 900 41EE0001

System: ACOR0010I 15.28.54 BEGIN DISPLAY
         00000900- 00000000 00000000 00000000 00000000 ........ ........
         00000910- 00000000 00000000 00000000 00000000 ........ ........
        ALTERED TO-
         00000900- 41EE0001 00000000 00000000 00000000 ........ ........
         00000910- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about system virtual memory (SVM).

ZACRS–Alter CRAS Status

Use this command to change the contents of the CRAS status table.

Requirements and Restrictions

None.

Format

```
►►──ZACRS── ──┬─REP─┬── ──┬─PRC─┬── ──┬─lniata─┬── ──TYPE-──┬─CON─┬─────────────►
              ├─FBK─┤     ├─RO──┤     └─leid───┘             ├─CRT─┤
              ├─DEL─┤     └─Axx─┘                            ├─PRT─┤
              └─CHG─┘                                        └─TYP─┘

►─ ──CPUID-id──┬──────────────────────────────────────────────────┬──────────────►
               │                                      (1)         │
               └── ──RTCD-──┬─PRC─────┬── ──OPT-──┬─VAL──────────┬┘
                            ├─RO──────┤           │   (2)        │
                            ├─TAPE────┤           ├─ADD──────────┤
                            ├─DASD────┤           │   (2)        │
                            ├─AUDT────┤           └─DEL──────────┘
                            ├─COMM────┤
                            ├─RDBS────┤
                            └─number──┘

►──┬──────────────┬──┬─────────────────────────────────────────────────┬────────►
   └── ──PRT-─Axx─┘  │         (3)                                     │
                     └── ──REST-──────┬─PRC─────┬── ──OPT-──┬─VAL─┬────┘
                                      ├─RO──────┤           ├─ADD─┤
                                      ├─TAPE────┤           └─DEL─┘
                                      ├─DASD────┤
                                      ├─AUDT────┤
                                      ├─COMM────┤
                                      ├─RDBS────┤
                                      └─number──┘

►──┬─────────────────────┬──────────────────────────────────────────────────────►◄
   │               (4)   │
   └── ──SUB-address─────┘
```

Notes:

1 The VAL option is valid only when you specify the FBK parameter.

2 The ADD and DEL options are valid only when you specify the CHG parameter.

3 The REST parameter is valid only for input devices.

4 The SUB parameter is valid only when you specify the FBK parameter and when 3270 native console support is generated.

REP
adds an entry to the CRAS table or replaces an existing entry.

FBK
performs fallback for the current 1052 PRC terminal to the alternate 1052 PRC terminal for 1052 console support, or performs fallback from the current PRC/RO terminal to an alternate PRC/RO terminal for 3270 native console support.

DEL
deletes an entry from the CRAS table.

CHG
modifies the current FSC routing codes or the terminal restrictions of an existing entry.

Note: The FSC routing codes are defined by the RTCD parameter, and the terminal restrictions are defined by the REST parameter.

PRC
specifies the prime CRAS.

RO
specifies the receive-only CRAS.

A*xx*
specifies an alternate CRAS, where *xx* is the 2-digit number of an alternate CRAS.

lniata
is a line, interchange, and terminal address.

leid
is a logical end-point identifier.

TYPE
specifies the terminal type.

CON
specifies a 1052/3215 type device.

CRT
specifies a CRT display device.

PRT
specifies a printer device.

TYP
specifies a teletypewriter.

CPUID-*id*
is a 1-character alphanumeric CPU ID.

Notes:

1. When you specify the CPUID parameter with the REP parameter, *id* is the ID of the emulator program (EP) processor, which owns the device.
2. When you specify the CPUID parameter with the CHG parameter, *id* is the ID of the processor from which messages will be routed to the emulator program (EP) processor, which owns the device.

RTCD
adds or deletes a functional support console (FSC).

PRC
specifies a prime CRAS FSC.

RO
specifies a receive-only FSC.

TAPE
specifies a tape FSC.

DASD
specifies a DASD FSC.

AUDT
specifies an audit trail FSC.

COMM
specifies a communications FSC.

RDBS
specifies a real-time database services (RDBS) FSC, which is used by the TPF Application Requester (TPFAR) feature and the recoup utility.

number
specifies the number of an FSC from 1–16.

OPT
specifies optional processing.

VAL
validates the console LNIATA.

In a 1052 environment, a test message is sent to the alternate 1052 prime CRAS and its entry is validated in the line status table.

In a native console support (NCS) environment, line 00 (RO) or line 01 (PRC) is validated in the line status table. This is necessary to recover line 00 or line 01 after a hardware failure occurred on a console and that console automatically fell back to an alternate CRAS device, causing the console to be marked as not valid in the line status table. After the line is validated, you can specify the REP parameter to return the PRC or RO to line 010000 or 000000, respectively.

ADD
adds optional data for the specified CRAS.

DEL
deletes optional data for the specified CRAS.

PRT-A*xx*
assigns the printer in the specified CRAS slot to the CRT display device in the specified CRAS, where *xx* is the CRAS slot from 1–99.

REST-
authorizes the CRAS terminal to issue commands that are restricted to the specified FSC.

SUB-*address*
is a subchannel address that is used for the PRC or RO fallback device.

Additional Information

- The parameters can be specified in any order.
- If a printer is deleted from the CRAS table, all references to it, which are assigned using the PRT parameter, are also deleted.
- The current PRC and RO can be replaced only with a terminal that was previously defined as an alternate set in the CRAS table. You can replace the PRC or RO by specifying the REP parameter.
- A 1052 PRC can be replaced with a CRT display device only if the CRT display device has an associated hardcopy device. You can associate a hardcopy device by specifying the PRT parameter.
- Enter the ZDCRS command to display CRAS status.

- If you are in a loosely coupled environment, the CHG parameter can only be entered from an EP processor.

Examples

An alternate CRAS is assigned in the following example.

```
User:   ZACRS REP A01 510000 CPUID-B TYPE-PRT

System: ACRS0000I 14.12.13 COMPLETED
```

In the following example, an alternate CRT display device is assigned with the printer defined in the previous example.

```
User:   ZACRS REP A02 4E0000 CPUID-B TYPE-CRT PRT-A01

System: ACRS0000I 14.12.19 COMPLETED
```

An RO device is replaced in the following example.

```
User:   ZACRS REP RO 510000 CPUID-B TYPE-PRT

System: ACRS0000I 14.16.16 COMPLETED
```

A PRC is replaced with a CRT display device in the following example.

```
User:   ZACRS REP PRC 4E0000 CPUID-B TYPE-PRT

System: ACRS0000I 14.19.23 COMPLETED
```

An alternate CRAS is assigned as a printer device on the emulator program (EP) processor B, and DASD-type messages are routed from processor C to that printer in the following example.

```
User:   ZACRS REP A01 510000 CPUID-B TYPE-PRT

System: ACRS0000I 14.12.13 COMPLETED

User:   ZACRS CHG A01 510000 CPUID-C TYPE-PRT RTCD-DASD OPT-ADD

System: ACRS0000I 14.22.11 COMPLETED
```

A printer is assigned as a functional support console in the following example.

```
User:   ZACRS CHG A01 510000 CPUID-B TYPE-PRT RTCD-DASD OPT-ADD

System: ACRS0000I 14.22.11 COMPLETED
```

A 3270 CRT display device is assigned as a functional support console in the following example.

```
User:   ZACRS CHG A02 4E0000 CPUID-B TYPE-CRT RTCD-DASD OPT-ADD

System: ACRS0000I 14.23.13 COMPLETED
```

A 3270 CRT functional support console is deleted in the following example.

```
User:   ZACRS CHG A02 4E0000 CPUID-B TYPE-CRT RTCD-DASD OPT-DEL

System: ACRS0000I 14.24.10 COMPLETED
```

A printer functional support console is deleted in the following example.

```
User:   ZACRS CHG A01 510000 CPUID-B TYPE-PRT RTCD-DASD OPT-DEL

System: ACRS0000I 14.24.46 COMPLETED
```

A CRAS terminal is given authorization to input DASD commands in the following example.

```
User:   ZACRS CHG A02 4E0000 CPUID-B TYPE-CRT REST-DASD OPT-ADD

System: ACRS0000I 14.27.01 COMPLETED
```

The CRAS terminal can no longer input DASD commands in the following example.

```
User:   ZACRS CHG A02 4E0000 CPUID-B TYPE-CRT REST-DASD OPT-DEL

System: ACRS0000I 14.27.39 COMPLETED
```

An alternate 1052 PRC is validated in the following example.

```
User:   ZACRS FBK PRC 010000 CPUID-B TYPE-CON OPT-VAL

System: ACRS0027I 15.42.43 ALT 1052 VALIDATED
```

A 1052 PRC is defined to fall back to the alternate 1052 PRC in the following example. Note that you must first validate the alternate 1052 PRC before you enter this command.

```
User:   ZACRS FBK PRC 010000 CPUID-B TYPE-CON

System: ACRS0045W 14.31.11 CONSOLE ALTERED, 1052 SUB CHANNEL 01F IS NOW THE PRIME CRAS.
```

An entry is removed from the CRAS table in the following example.

```
User:   ZACRS DEL A01 510000 CPUID-B TYPE-PRT

System: ACRS0000I 14.36.05 COMPLETED
```

A 3270 native console is defined to fall back to the specified fallback console in the following example.

```
User:   ZACRS FBK PRC 010000 CPUID-B TYPE-CRT SUB-023 OPT-VAL

System: ACRS0045W 14.38.40 CONSOLE ALTERED - THIS IS NOW PRC
```

A 3270 native console is defined to fall back to any fallback alternate console in the following example.

```
User:   ZACRS FBK PRC 010000 CPUID-B TYPE-CRT

System: ACRS0045W 14.39.02 CONSOLE ALTERED, 3270 SUB CHANNEL 022 IS NOW THE PRIME CRAS.
```

Related Information

See *TPF Data Communications Services Reference* for more information about CRAS support.

ZADCA–Alter Dump Label

Use this command to change as many as 16 bytes of storage by specifying the address of a dump label.

Requirements and Restrictions

You must specify the IS parameter in a tightly coupled system when prefixing applies; for example, when you change storage in page 0 of an I-stream, or when you change the I-stream unique global areas.

Format

```
►►──ZADCA── ──dumplabel── ──rsa── ──newdata──┬─────────────────────┬──────────►
                                             └─ ──Valdata-olddata──┘

    ┌─ ──Is-1──┐
►───┴─ ──Is-xx─┴──────────────────────────────────────────────────────────────►◄
```

dumplabel
: is a 3-character alphanumeric dump label.

rsa
: is a 1- to 8-digit hexadecimal relative starting address (offset) from the dump label.

newdata
: is the new data that replaces the old data. The new data must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes).

Valdata-*olddata*
: verifies that *olddata* matches the data being changed. If there is a discrepancy, no data is changed. The *olddata* variable must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes). The length of *olddata* can be different from the length of *newdata*.

  Note: If you do not specify this parameter, the data is changed without any verification.

Is-*xx*
: specifies the I-stream that this change affects where, *xx* is a decimal number from 1 to 16.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZADCA HELP
  ZADCA ?
- The TPF system displays the contents of storage before and after you change it.
- Enter the ZDDCA command to display storage.

Examples

In the following example, 4 bytes of data beginning at the CLV dump label are changed on I-stream 1.

```
User:   ZADCA CLV 0 41EE0001

System: ADCA0010I 13.18.35 BEGIN DISPLAY
         00000BBC- 0096012C 0096012C 00000000 00000000 .o...o.. ........
         00000BCC- 00000000 00000000 00000000 00000000 ........ ........
        ALTERED TO- _
         00000BBC- 41EE0001 0096012C 00000000 00000000 .....o.. ........
         00000BCC- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

In the following example, 1 byte of data beginning at an offset of X'22' bytes from the CLV dump label is changed on I-stream 2.

```
User:   ZADCA CLV 22 FF I-2

System: ADCA0010I 13.18.35 BEGIN DISPLAY
         00000BBC- 00000010 00000194 91004140 40008400 .......m j.....d.
         00000BCC- 00000010 00000194 91004140 40008400 .......m j.....d.
        ALTERED TO- _
         00000BBC- 0000FF10 00000194 91004140 40008400 .......m j.....d.
         00000BCC- 00000010 00000194 91004140 40008400 .......m j.....d.
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about changing storage using a dump label.

ZAFIL–Alter File

Use this command to change as many as 32 bytes of a file record.

Requirements and Restrictions

- For file records that also reside in VFA, you must specify a file address reference format (FARF) address when it is necessary to change both the VFA copy and the file copy. If you specify an MCHR address, only the file copy of the record is changed. The VFA copy is deleted from VFA before the change is made.
- You cannot use this command to change general dataset records.

Format

```
►►──ZAFIL── ──┬─fileaddr───────┬── ──rsa── ──newdata──────────────────────────►
              ├─mmmmccccchhhhrr─┤
              └─mmggggggq──────┘

                                   ┌──Type-R─────┐
►──┬───────────────────┬──┼──────────────┼─────────────────────────────►◄
   └──Valdata-olddata─┘  └──Type-──┬─P─┤
                                      ├─D─┤
                                      └─G─┘
```

fileaddr
is an 8- or 16-digit hexadecimal file address.

Note: If you specify the file address using the FARF addressing format, the duplicate copy of the file is also updated.

mmmmccccchhhhrr
is a 14-digit extended MCHR file address, where:

mmmm
is the 4-digit hexadecimal symbolic module number.

cccc
is the 4-digit hexadecimal cylinder number.

hhhh
is the 4-digit hexadecimal head number.

rr is the 2-digit hexadecimal record number.

mmgggggq
is an 8-digit general file (GF) pseudo module number and relative record number, where:

mm
is the 2-digit hexadecimal symbolic GF module number.

ggggg
represents bits 8–27 of a GF relative record number as a binary counter.

q is a single hexadecimal digit that represents the low-order 4 bits of the GF relative record number as follows:

Bit	Description
28	Unit position of the relative record number.
29	Must be 1 to indicate a relative record number.
30	Must be 0.
31	The number 1 to indicate a large record, or the value 0 to indicate a small record.

rsa
is a 1- to 3-digit hexadecimal relative starting address (offset).

Note: If the relative starting address is not on a fullword boundary, it is automatically adjusted to the next lower fullword boundary.

newdata
is the new data that replaces the old data. The new data must be an even number of hexadecimal digits and cannot exceed 64 digits (32 bytes).

Valdata-*olddata*
verifies that *olddata* matches the data being changed. If there is a discrepancy, no data is changed. The variable *olddata* must be an even number of hexadecimal digits and cannot exceed 64 digits (32bytes). The length of *olddata* can be different from the length of *newdata*.

Note: If you do not specify this parameter, the data is changed without any verification.

Type
specifies which file copy of the record to change:

R retrieves and displays the prime or duplicate copy of the record and then changes both the prime and duplicate copies.

P retrieves and displays the prime copy of the record and then changes both the prime and duplicate copies.

D retrieves and displays the duplicate copy of the record and then changes both the prime and duplicate copies.

G changes the 4-byte pseudo module number and relative record number copy.

Note: The P and D options are valid only for FARF format addresses.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZAFIL HELP
  ZAFIL ?
- Enter the ZCSON command to convert MCHR addresses to FARF3, FARF4, FARF5, and FARF6 format depending on the system migration stage and dispense mode.

Examples

A file record is verified before it is changed in the following example. Notice that the data is displayed before and after it is changed.

```
User:   ZAFIL 0000000038880007 3 20 VALDATA-80

System: AFIL0011I 07.10.02 DISPLAY OF FILE ADDRESS 0000000038880007
         00000000- C1D60080 C3C7E3F2 00000514 00000515 AO..CGT2 ........
         00000010- 00020FE8 0000432C 00004EB4 000023D0 ...Y.... ........
        ALTERED TO-
         00000000- C1D60020 C3C7E3F2 00000514 00000515 AO..CGT2 ........
         00000010- 00020FE8 0000432C 00004EB4 000023D0 ...Y.... ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

A general file record is changed in the following example.

```
User:   ZAFIL 0D000005 8 01 T-G

System: AFIL0011I 07.42.23 DISPLAY OF FILE ADDRESS 000000000D000005
         00000008- 00000000 00404040 0D000085 0D00033D .....    ...e....
         00000018- 0D0005F5 00000000 00000000 004E0000 ...5.... ........
        ALTERED TO-
         00000008- 01000000 00404040 0D000085 0D00033D .....    ...e....
         00000018- 0D0005F5 00000000 00000000 004E0000 ...5.... ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about changing file records.

ZAGFL–Alter General File Label Track

Use this command to change the label of a general file.

Requirements and Restrictions

The specified device must be mounted before you enter this command.

Format

```
►►──ZAGFL── ──datasets── ──address──────────────────────────────────►◄
```

datasets
: are 3 groups of 2-digit data set numbers followed by the 4-character alphanumeric name of the subsystem; for example, `030405ABCD`.

Note: You must include all trailing blanks for the subsystem name.

address
: is a 3- to 4-digit hexadecimal device address.

Additional Information

Enter the ZDGFL command to verify the change. (The changed information is not displayed when you enter this command.)

Examples

The label of the specified general file is changed in the following example. Notice that the required trailing blank for the subsystem ID is included in the command.

```
User:   ZAGFL 030405RTM  0292

System: AGFL0024I 15.24.02 GENERAL FILE DATA SETS   -  -   ARE ON DEVICE 0292, SUBSYSTEM 0003
        AGFL0023I 15.24.02 ALTERATION COMPLETE
```

Related Information

See *TPF Database Reference* for more information about general file support.

ZALCT–Alter Line Path Status

Use this command to turn on or turn off one or more line paths in the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZALCT── ──LN── ──line── ──┬─ACT─┬── ──┬─path─┬──────────────────────────►◄
                              └─DWN─┘     └─ALL──┘
```

line
is a 2-digit hexadecimal symbolic line number.

ACT
turns on the path.

DWN
turns off the path.

path
is a 2-digit hexadecimal symbolic path number.

ALL
turns on or turns off all the paths for the specified line.

Additional Information

Enter the ZDLCT command to display line path status.

Examples

All paths on the specified line are turned off in the following example.

```
User:   ZALCT LN 3E DWN ALL

System: 11.49.25 ALCT LN 3E AL OFF OK
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZALMT–Acquire NEF LMT Queue Ownership

Use this command to obtain ownership of the long message transmitter (LMT) queue for the current processor from another processor.

Requirements and Restrictions

None.

Format

```
►►──ZALMT── ──FROM-cpuid──┬───────┬──────────────────────────────────►◄
                          └─ ──BP─┘
```

cpuid
: is the 1-character alphanumeric CPU ID of the loosely coupled processor from which this processor will obtain LMT ownership.

BP
: bypasses system state checks.

Additional Information

None.

Examples

In the following example, ownership of the LMT queue is obtained from processor C.

```
User:   ZALMT FROM-C

System: ALMT0001I 13.20.40  CPU C NEF LMT QUEUE ACQUIRED BY CPU B
```

Related Information

See *TPF Data Communications Services Reference* for more information about the long message transmitter.

ZAMAP–Online Map File Load and Update

Use this command to load terminal map records to the TPF system from the map macro tape file (MRT). The MRT is used as input to the online portion of the map file create and load program.

Requirements and Restrictions

You can enter this command only when the system is in NORM state.

Format

```
►►──ZAMAP──────────────────────────────────────────────────────────►◄
```

Additional Information

- The mapping and formatting function is suspended until the phase of processing started by ZAMAP ends.
- Any errors that are detected during the load phase are recorded on the prime CRAS (PRC) high-speed read-only (RO) device.

Examples

```
User:   ZAMAP

System: MAINT COMPLETE
```

Related Information

See *TPF Data Communications Services Reference* for additional information about online map file load and update.

ZAMOD–Alter DASD Status

Use this command to change the status of a direct access storage device (DASD) from online to offline, and offline to online. This command updates the module file status table (MFST) and module cross-reference table to reflect the configuration change. A duplicate update also starts when the DASD is brought online.

Using ZAMOD?

ZMCPY UP or ZMCPY DOWN are the preferred commands. The ZAMOD command, while still supported, actually issues a ZMCPY UP or ZMCPY DOWN command internally. Online messages that are returned by the TPF system indicate that a ZMCPY command was entered.

Note: In some places, DASD is also referred to as module.

Requirements and Restrictions

None.

Format

```
►►──ZAMOD── ──module── ──device── ──┬─U─┬──────────────────────────────────────►◄
                                    └─D─┘
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time packs or the pseudo module number for general files.

device
: is a 3- to 4-digit hexadecimal device address.

U places the DASD online (up).

D places the DASD offline (down).

Additional Information

- You can use this command to bring a DASD online if it was taken offline by a command or during system error recovery.
- Enter the ZMCPY ABORT command to stop the ZAMOD UP function.

Examples

The status of the specified DASD is allowed to change from online to offline in the following example.

```
User:   ZAMOD 4A EE0 D

System: MCPY0294I 14.46.36 OK MOD 04A OFF DEVICE 0EE0
```

Related Information

See *TPF Database Reference* for more information about module up and module down processing.

ZAPAT–Alter Program Allocation Table

Use this command to change the allocation parameters of a program in the file copy or the core (main storage) copy of the program allocation table (PAT).

If you change the file copy of the PAT:

- The changes apply to the prime module version.
- The changes do not take effect until the next initial program load (IPL) is performed on the TPF system.
- The changes do not take effect if an IPL is performed on the loader general file.

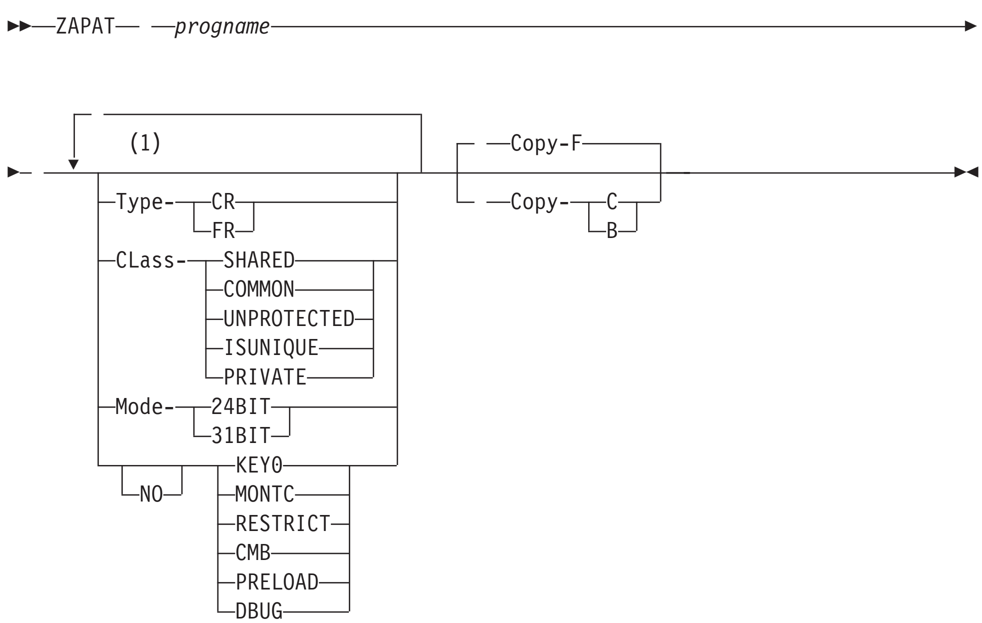
If you change the core copy of the PAT:

- The changes apply to the allocated version of the program and all versions of the program in core that were activated using the E-type loader.

Requirements and Restrictions

- You can only change the PAT for the image that is currently active.
- ISO-C programs, dynamic load modules (DLMs), and libraries must be assigned SHARED class.
- C load modules are treated as core resident, regardless of how you allocate them.

Format



Notes:

1 You can also separate the parameters with a comma (,).

progname
is a 4-character alphanumeric program name.

Type
changes the residency of the program:

CR
specifies that the program is core resident.

FR
specifies that the program is file resident.

This parameter is valid only when changing the file copy of the PAT.

CLass
changes the residency characteristics of a program to one of the following classes. See Table 2 on page 74 for a summary of the residency characteristics that can be assigned to core resident and file resident programs.

This parameter is valid only when changing the file copy of the PAT.

SHARED
specifies that the program is a shared program. That is, all ECBs share the same key-protected copy of the program, but not all ECBs see the program at the same address. In addition, ECBs that are not currently using the program may not be able to see the program.

Both core resident and file resident programs can be assigned SHARED class.

COMMON
specifies that the program is a common program. That is, all ECBs share the same key-protected copy of the program and all ECBs see the program at the same address.

Both core resident and file resident programs can be assigned COMMON class.

UNPROTECTED
specifies that the program is an unprotected program. That is, all ECBs share the same unprotected storage copy of the program, and all ECBs see the program at the same address. This class is intended for self-modifying programs.

Only file resident programs can be assigned UNPROTECTED class.

ISUNIQUE
specifies that the program is an I-stream unique program. That is, all ECBs using the program on a specific I-stream share the same unprotected storage copy of the program, and all ECBs see the program at the same address. This class is intended for self-modifying, I-stream unique programs.

Only file resident programs can be assigned ISUNIQUE class.

PRIVATE
specifies that the program is a private program. That is, an ECB is supplied with a new storage copy of the program each time the ECB enters the program.

Only file resident programs can be assigned PRIVATE class.

Mode
changes the addressing mode in which the program is entered:

24BIT
specifies that the program is entered in 24-bit addressing mode.

31BIT
specifies that the program is entered in 31-bit addressing mode.

This parameter is valid only when changing the file copy of the PAT.

Copy
specifies which version of the PAT to change:

F changes the file copy.

C changes the core copy.

B changes both the core and the file copy.

See Table 3 on page 74 for a summary of the allocation parameters that can be changed in the file copy and the core copy of the PAT.

NO
turns off the following allocation parameters.

KEY0
specifies that the program can change the protection key to 0 using the CINFC W macro. This parameter is valid when you change both the core and the file copy of the PAT.

MONTC
specifies that the program can enter supervisor state using the MONTC macro. This parameter is valid when you change both the core and the file copy of the PAT.

RESTRICT
specifies that the program can issue restricted macros. This parameter is valid when you change both the core and the file copy of the PAT.

CMB
specifies that the program can obtain common storage blocks. This parameter is valid when you change both the core and the file copy of the PAT.

PRELOAD
specifies that the program is brought into main storage before 1052 state. This parameter is valid when you change both the core and the file copy of the PAT.

Notes:

1. Only core resident programs can be assigned the PRELOAD allocation parameter.
2. The number of programs that are assigned the PRELOAD allocation parameter should be minimized for performance reasons.

DBUG
specifies that the program is available to be debugged using the TPF Assembler Debugger for VisualAge Client. This parameter is valid when you change both the core and the file copy of the PAT. All programs are assumed to be available for debugging unless NODBUG is specified. The NODBUG parameter is available only for programs that are allocated as 31-bit core resident or file resident with a class of SHARED or COMMON.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZAPAT HELP
  ZAPAT ?

- The COPY parameter is the only parameter that has a default value. That is, if you do not specify a value for the COPY parameter, the TPF system assumes the default value. If you do not specify a value for the other parameters, the TPF system assumes the current value.
- If the program you are changing is a transfer vector, the allocation parameters of the parent program are changed rather than the allocation parameters of the transfer vector.
- Enter the ZDPAT command to display the PAT.
- Table 2 summarizes the residency characteristics that can be assigned to core resident and file resident programs.

Table 2. Summary of the CLASS Allocation Parameter

Class	Core Resident	File Resident
SHARED	Yes	Yes
COMMON	Yes	Yes
UNPROTECTED	No	Yes
ISUNIQUE	No	Yes
PRIVATE	No	Yes

- Table 3 summarizes which allocation parameters can be changed in the file copy and the core copy of the PAT.

Table 3. Summary of the COPY Allocation Parameter

Parameter	COPY-F	COPY-C	COPY-B
TYPE	Yes	No	No
CLASS	Yes	No	No
MODE	Yes	No	No
KEY0	Yes	Yes	Yes
NOKEY0	Yes	Yes	Yes
MONTC	Yes	Yes	Yes
NOMONTC	Yes	Yes	Yes
RESTRICT	Yes	Yes	Yes
NORESTRICT	Yes	Yes	Yes
CMB	Yes	Yes	Yes
NOCMB	Yes	Yes	Yes
PRELOAD	Yes	Yes	Yes
NOPRELOAD	Yes	Yes	Yes
DBUG	Yes	Yes	Yes
NODBUG	Yes	Yes	Yes

Examples

In the following example, the program residency of the CVAA program is changed from shared to common. IMAGEABC is the active image.

```
User:   ZAPAT CVAA CLASS-COMMON

System: APAT0101I 11.34.30  BEGIN DISPLAY OF FILE COPY FOR IMAGE IMAGEABC
         PROGRAM          CVAA

         VERSION          JW
         TYPE             FILE RESIDENT
         BASE PAT SLOT    0172DA80
         LINKAGE TYPE     TAR
         CLASS            SHARED
         FILE ADDRESS     2F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    NONE

        ALTERED TO -
         PROGRAM          CVAA

         VERSION          JW
         TYPE             FILE RESIDENT
         BASE PAT SLOT    0172DA80
         LINKAGE TYPE     TAR
         CLASS            COMMON
         FILE ADDRESS     2F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    NONE


        DISPLAY OF PAT SLOTS FOR CVAA

        VERSN LOADSET  ACT NUM STATUS      FILE ADDR TYPE LINK PAT ADDR
        ----- -------- ------- ----------- --------- ---- ---- --------
          TP  MKING        13  SELECT ACT  3F801CFE   FR  BAL  01567018
          TP  TPUENTE       6  ACTIVE      3FD809A7   FR  TAR  01635B80
          JW  BASE          0  ACTIVE      2F800CF7   FR  TAR  0172DA80
        END OF DISPLAY
```

In the following example, the KEY0 and MONTC allocation parameters are changed in both the core and file copy of the PAT. The CVIC program can now change the protection key to 0 and enter supervisor state.

Note: Because CVIC is a transfer program of CVIA, the allocation parameters of the parent program (CVIA) are changed rather than the allocation parameters of CVIC. There are three versions of CVIA active: the base version and two versions that were activated by the E-type loader.

```
User:   ZAPAT CVIC COPY-B KEY0 MONTC

System: APAT0101I 11.34.30  BEGIN DISPLAY OF FILE COPY FOR IMAGE IMAGEABC
         PROGRAM          CVIA PARENT OF CVIC

         VERSION          MG
         TYPE             CORE RESIDENT
         BASE PAT SLOT    0172EB80
         LINKAGE TYPE     TAR
         CLASS            COMMON
         FILE ADDRESS     3F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    NONE

        ALTERED TO -
         PROGRAM          CVIA PARENT OF CVIC

         VERSION          MG
         TYPE             CORE RESIDENT
         BASE PAT SLOT    0172EB80
         LINKAGE TYPE     TAR
         CLASS            COMMON
         FILE ADDRESS     3F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    KEY0 MONTC

        BEGIN DISPLAY OF CORE COPY FOR IMAGE IMAGEABC
         PROGRAM          CVIA PARENT OF CVIC

         VERSION          MG
         TYPE             CORE RESIDENT
         BASE PAT SLOT    0172EB80
         LINKAGE TYPE     TAR
         CLASS            COMMON
         FILE ADDRESS     3F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    NONE

        ALTERED TO -
         PROGRAM          CVIA PARENT OF CVIC

         VERSION          MG
         TYPE             CORE RESIDENT
         BASE PAT SLOT    0172EB80
         LINKAGE TYPE     TAR
         CLASS            COMMON
         FILE ADDRESS     3F800CF7
         ADDRESSING MODE  31BIT
         AUTHORIZATION    KEY0 MONTC

        DISPLAY OF PAT SLOTS FOR CVIA

        VERSN LOADSET  ACT NUM STATUS      FILE ADDR TYPE LINK PAT ADDR
        ----- -------- ------- ----------- --------- ---- ---- --------
          IK  GRATEFUL     13  ACTIVE      3FD80777   CR  BAL  01568018
          JG  JGARCIA       6  SELECT ACT  3FD807A7   CR  TAR  0166BCC0
          MG  BASE          0  ACTIVE      3F800CF7   CR  TAR  0172EB80
        END OF DISPLAY
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Installation Support Reference* for more information about the program allocation table (PAT).

ZAPGM–Alter Program

Use this command to change as many as 16 bytes of a program starting at a specified program listing address. You can change real-time programs (both file resident and core resident), control programs, and core image restart area programs (FCTB, ACPL, ICDF, IPLB, SIGT, RIAT, and IPAT in the core resident program area).

Requirements and Restrictions

- The LOADSET parameter and the *version* variable are valid only after the E-type loader restart routine completes.
- This command cannot be used to change the following programs:
  - Core-copy private programs.
  - Loader general file (LGF) copy of the control program (CP). Changes made to the CP are made only to the prime module version of the CP.
  - Core-copy, I-stream unique programs. To change an I-stream unique program, use the ZRPGM command to lock the program in core and then use the ZACOR command to change it.
  - The core copy of ACPL, IPLA, and IPLB cannot be changed because the core copy is in storage only at restart time.
- This command cannot be entered for the file copies of C load modules.

Format

```
►►──ZAPGM── ──progname──┬──────────┬── ──rsa── ──newdata──────────────────────►
                        └─.version─┘

                                                          ┌─ ──Copy-F──────┐
►──┬──────────────────┬──┬─────────────────────────┬──────┼────────────────┼──►
   └─ ──Valdata-olddata┘  └─ ──Loadset-┬─lsname─┬──┘      └─ ──Copy-┬─C─┬──┘
                                       └─BASE───┘                  └─B─┘

   ┌─ ──Type-R──────┐
►──┼────────────────┼─────────────────────────────────────────────────────►◄
   └─ ──Type-┬─P─┬──┘
             └─D─┘
```

progname
: is a 4- to 6-character alphanumeric program name.

version
: is a 2-character alphanumeric program version code. This parameter can be specified only for real-time programs.

rsa
: is a 1- to 6-digit hexadecimal relative starting address (offset) in the program listing.

newdata
is the new data that replaces the old data. The new data must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes).

Valdata-*olddata*
verifies that *olddata* matches the data being changed. If there is a discrepancy, no data is changed. The *olddata* variable must be an even number of hexadecimal digits and cannot exceed 32 digits (16 bytes). The length of *olddata* can be different from the length of *newdata*.

Notes:

1. If you do not specify this parameter, the data is changed without any verification.
2. If you specify this parameter when you specify the B option for the COPY parameter, an error will occur if *olddata* is not the same in both the file copy and the core copy.

Loadset
specifies the loadset that contains the program you want to change.

lsname
is the 5- to 8-character alphanumeric name of a loadset.

BASE
indicates that you want to change the base version of the program. You can specify this parameter only for real-time programs.

Copy
specifies which copy of the program to change:

F changes the file copy.

C changes the core (main storage) copy.

B changes both the core and the file copy.

Type
specifies which file copy of the program to change:

R retrieves and displays either the prime or duplicate copy of the program and then changes both the prime and duplicate copies.

P retrieves and displays the prime copy of the program and then changes both the prime and duplicate copies.

D retrieves and displays the duplicate copy of the program and then changes both the prime and duplicate copies.

Note: This parameter is ignored if you specify the C option for the COPY parameter.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZAPGM HELP
  ZAPGM ?
- Enter the ZDPGM command to display data from a program.
- You can change programs that are in the process of being accepted by specifying the BASE option for the LOADSET parameter. You can also change the current base version of a program while a new program is in the process of being accepted by specifying the version code of the current base program.

- If you change the core copy of a file that resides in virtual file access (VFA) memory, the program is copied to a common block before it is changed. ECBs that are entered before the program is changed continue to use the original VFA version of the program. ECBs that are entered after the program is changed use the new version of the program in the common block.
- If you change a real-time program in the core resident program area (CRPA) and increase its size, the program is copied to a common block before it is changed. The core copies of C load modules cannot be altered in size. ECBs that are entered before the program is changed continue to use the original CRPA version of the program. ECBs that are entered after the program is changed use the new version in the common block. The area in the CRPA that contains the original version of the program is not returned to the TPF system until the next initial program load (IPL) is performed.
- If you change the core copy of a real-time program, it is first locked in core. This ensures that the changed program remains in core.
- If you do not specify a program version code or a loadset name, and there is more than one version of the program active in core, the activation number of the ECB determines the version of the program that is changed.
- You can change the file copy of a real-time program in a loadset even if the loadset is not active.
- You can change the core copy of a real-time program in a loadset that was deactivated if the PAT slot for the program version was not deleted. However, if the loadset is activated again, you cannot change the old core copy of the real-time program in the loadset because all references to that loadset use the copy of the program that was activated in core.
- The ZAPGM command changes program records for the image that is currently used by the processor. It cannot be used to change the real-time program records for other images. If a CIMR component on the current image logically references another image, the logical copy is changed. In this case, the response message indicates the actual image that was changed.
- If the TPF system is in patch mode, only core copy updates are allowed. For more information about patch mode, see “ZPTCH–Maintain Memory Patch Decks” on page 1136.
- If you enter the ZAPGM command before the E-type loader restart routine ends, you cannot specify the LOADSET parameter or the *version* variable. Therefore, you can use this command to change only the base version of a program.
- If you use the ZAPGM command to change the core copy of the control program (CCNUCL) below X'1000', which is the prefix page, remember the following:
  - The prefix page of the control program is unique per processor (I-stream).
  - The prefix page of the control program is changed on all processors in a tightly coupled environment.
  - If you specify the VALDATA parameter, the data that you specify is compared to the data in the prefix page of the control program on ***each*** processor. If there is a discrepancy on ***any*** processor, no data is changed.
  - Only the changes made to the main I-steam prefix page are displayed; the changes made to the other prefix pages are not displayed.
- If you use the ZAPGM command to change a real-time program that has the ADATA file loaded, the assembler debugger may not display an accurate listing view of the program.

  See *TPF System Installation Support Reference* for more information about loading ADATA files.

Examples

The file copy of a real-time program is changed in the following example.

```
User:   ZAPGM CVAI 8 00404040

System: APGM0010I 10.48.54 BEGIN DISPLAY OF FILE COPY FOR
                   CVAI.40 ACTIVE IN LOADSET BASE
         00000008- 00000011 000001AC 91004140 40008400 ........ j..  .d.
         00000018- C5C3C2D3 4DE0B204 0000D104 00000000 ECBL(... ..J.....
        ALTERED TO-
         00000008- 00404040 000001AC 91004140 40008400 .    .... j..  .d.
         00000018- C5C3C2D3 4DE0B204 0000D104 00000000 ECBL(... ..J.....
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Both the file and core copy of a control program are changed in the following example.

```
User:   ZAPGM CC3705 10 FFFF VALDATA-00 COPY-B

System: APGM0010I 15.50.37 BEGIN DISPLAY OF FILE COPY FOR
                 CC3705 VERSION IY IN IMAGE IMAGEABC
         00000010- 00000000 00000000 001A0000 00C00000 ........ ........
         00000020- 00000000 00000000 00020010 1B4458A0 ........ ........
        ALTERED TO-
         00000010- FFFF0000 00000000 001A0000 00C00000 ........ ........
         00000020- 00000000 00000000 00020010 1B4458A0 ........ ........
                 BEGIN DISPLAY OF CORE COPY FOR
                 CC3705 VERSION IY
         00000010- 00000000 00000000 001A0000 00C00000 ........ ........
         00000020- 00000000 00000000 00020010 1B4458A0 ........ ........
        ALTERED TO-
         00000010- FFFF0000 00000000 001A0000 00C00000 ........ ........
         00000020- 00000000 00000000 00020010 1B4458A0 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about changing program data.

ZAREC–Alter a Fixed File or Pool Record

Use this command to change as many as 32 bytes of a fixed file record or pool record starting at a specific relative starting address. You can specify the record type as a hexadecimal number, SYSEQ label, or pool type.

Requirements and Restrictions

None.

Format

```
►►──ZAREC── ──┬─rectype─┬─.ordnum── ──rsa── ──newdata──────────────────────────►
              ├─Llabel──┤
              └─Ppool───┘

                               ┌ ──Type-R────────┐  ┌ ──Is-1──┐
►──┬───────────────────────┬───┼─────────────────┼──┼─────────┼──┬──────────────┬──►
   └ ──Valdata-olddata─────┘   └ ──Type-──┬─P─┬──┘  └ ──Is-xx─┘  └ ──Cpuid-id───┘
                                          └─D─┘

►──┬───────────────┬────────────────────────────────────────────────────────────►◄
   └ ──Ssu-user────┘
```

rectype
: is a 1- to 4-digit hexadecimal record type (for FACE).

L*label*
: is a 1- to 8-character SYSEQ label (for FACS).

Note: The number sign (#) that precedes the SYSEQ label is optional unless it is needed to distinguish between 2 labels; for example, #RECA and RECA.

P*pool*
: is a pool record, where *pool* is one of the following pool record types:

SLT
: Small, long-term

SST
: Small, short-term

SDP
: Small, long-term duplicate

LLT
: Large, long-term

LST
: Large, short-term

LDP
: Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

ordnum
is a 1- to 16-digit hexadecimal ordinal number.

rsa
is a 1- to 3-digit hexadecimal relative starting address (offset) in the record.

newdata
is the new data that replaces the old data. The new data must be an even number of hexadecimal digits and cannot exceed 64 digits (32 bytes).

Valdata-*olddata*
verifies that *olddata* matches the data that is being changed. If there is a discrepancy, no data is changed. The *olddata* variable must be an even number of hexadecimal digits and cannot exceed 64 digits (32 bytes). The length of *olddata* can be different from the length of *newdata*.

Note: If you do not specify this parameter, the data is changed without any verification.

Type
specifies which copy of the record to retrieve:

R retrieves and displays either the prime or duplicate copy of the record and then changes both the prime and duplicate copies.

P retrieves and displays the prime copy of the record and then changes both the prime and duplicate copies.

D retrieves and displays the duplicate copy of the record and then changes both the prime and duplicate copies.

Is-*xx*
specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Cpuid-*id*
is the 1-character alphanumeric CPU ID of the processor.

Ssu-*user*
is a subsystem user.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZAREC HELP
  ZAREC ?
- Enter the ZDREC command to display data from a record.

Examples

In the following example, data in the specified fixed file record is changed.

```
User:   ZAREC 023.000009 10C D1C6D2

System: AREC0011I 10.48.44 DISPLAY OF FILE ADDRESS 000000001180004F
         0000010C- D3C1E7A8 18210000 00000000 00000000 LAXy.... ........
         0000011C- FE1443C2 D3C1E76A 18A10000 00000000 ...BLAX. .s......
        ALTERED TO-
         0000010C- D1C6D2A8 18210000 00000000 00000000 JFKy.... ........
         0000011C- FE1443C2 D3C1E76A 18A10000 00000000 ...BLAX. .s......
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

In the following example, the data that is changed is verified before it is replaced by the new data.

```
User:   ZAREC LRV1RI.4 192 D5C5E3C2 VALDATA-40404040

System: AREC0011I 10.48.44 DISPLAY OF FILE ADDRESS 0000000042000027
         00000190- 00004040 40404040 4040C3C9 C3E2E7C2 ..         CICSXB
         000001A0- 40400000 80000000 00000000 00000000   ...... ........
        ALTERED TO-
         00000190- 0000D5C5 E3C24040 4040C3C9 C3E2E7C2 ..NETB     CICSXB
         000001A0- 40400000 80000000 00000000 00000000   ...... ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

A pool record is changed in the following example.

```
User:   ZAREC PLST.40 10 D1C6D2A8

System: AREC0011I 13.34.52 DISPLAY OF FILE ADDRESS 0000000000040101
         00000010- 00000000 00000000 00000000 00000000 ........ ........
         00000020- 00000000 00000000 00000000 00000000 ........ ........
        ALTERED TO-
         00000010- D1C6D2A8 00000000 00000000 00000000 JFKy.... ........
         00000020- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Concepts and Structures* for more information about fixed file records and pool records.

ZASER–Alter System Error Options

Use this command to change the system error options and the SNAP dump options. These options control the amount and type of data written to the system console and the dump device when a system error dump (SERRC) or SNAP dump (SNAPC) occurs. These options also control the amount and type of data passed to the dump data user exit.

Requirements and Restrictions

None.

Format

```
                          ┌─ ─────────┐
                          ▼           │
►►──ZASER──┬────────┬─  ───option─────┴──────────────────────────────►◄
           └─ ─Snap─┘
```

Snap
: indicates that the specified options are SNAP dump options. If you do not specify the SNAP parameter, the specified options are assumed to be system error options.

option
: is one or more of the following system error or SNAP dump options:

Option	System Error	SNAP Dump
CONSALL		Causes the SNAP routines to write a message and as many as 1024 bytes of SNAP data to the system console whenever a SNAP error is generated. (SNAP data is the data identified by one or more LISTC macro calls associated with the SNAPC macro.)
CONSDMP	Causes the system error routine to log a message to the system console whenever a dump is generated. NODUMP messages are suppressed.	
CONSMSG		Causes the SNAP routines to log a message to the system console whenever a SNAP error is generated.
CONSOFF	Causes the system error routine to stop logging all error information to the system console.	Causes the SNAP routines to stop logging all error information to the system console.
CONSON	Causes the system error routine to log a message to the system console whenever an error occurs.	

Option	**System Error**	**SNAP Dump**
DATAX	Causes the system error routine to pass SNAPC and SERRC dump data to the dump data user exit.	
DUMPMSG		Causes SNAP messages to be written to the dump device.
DUMPOFF	Causes the system error routine to stop logging all error information to the dump device. NODUMP is substituted in place of the system dump number in the console message. No system error information is passed to the dump data user exit. Manual dumps are not affected.	Causes the SNAP routines to stop logging all error information to the dump device. No SNAPC data is passed to the dump data user exit.
DUMPON	Causes system error dumps to be written to the dump device. System error information is passed to the dump data user exit if the DATAX option is specified.	Causes SNAP dumps to be written to the dump device. SNAPC data is passed to the dump data user exit if the DATAX option is specified.
DUPL	If an ECB-controlled program generates the same system error more than once, each occurrence of the error generates an OPR dump.	If the same SNAP error is generated more than once, each occurrence of the error generates a SNAP dump.
LOG	Causes the system error routine to log a message to the real-time dump tape whenever an error occurs.	
LONG	Includes the information contained in SHORT and MEDIUM dumps in OPR dumps. Permanent storage is also included in dumps following the ECB working storage.	
MEDIUM	Includes information contained in SHORT dumps as well as the following information in OPR dumps: • Collated macro trace • Collated I/O trace • Branch trace • Information about other instruction streams • Storage areas around the registers • Failing instructions.	
NODATAX	Causes the system error routine to stop passing dump data to the dump data user exit.	

Option	System Error	SNAP Dump
NODUPL	If an ECB-controlled program generates the same system error more than once on the same subsystem, only the first occurrence of the error generates an OPR dump. Subsequent dumps are not generated and only the message is passed to the dump data user exit. NODUMP is substituted in place of the system dump number in the console message. **Note:** This option applies only to OPR dumps. System (CTL) and manual dumps are generated regardless of the setting of this option.	If the same SNAP error is generated more than once, only the first occurrence of the error generates a SNAP dump. Subsequent dumps are not generated.
NOLOG	Causes the system error routine to stop logging messages to the real-time dump tape.	
PRT	Causes all system error dumps to be written to the real-time printer when the in-core dump formatter (ICDF) is loaded to the TPF system.	Causes the SNAP dumps to be written to the real-time printer when ICDF is loaded to the TPF system.
PRT ECB CNTRLD	Causes OPR dumps to be written to the real-time printer when ICDF is loaded to the TPF system. System (CTL) dumps are still written to the real-time dump tape.	
SHORT	Includes only the ECB, the working storage associated with the ECB, and the prefix of the failing instruction stream in OPR dumps.	
TAPE	Causes system error dumps to be written to the real-time dump tape.	Causes the SNAP dumps to be written to the real-time dump tape.
THEAPON	Includes the heap storage of the entire process if the ECB is running in a threaded environment.	
THEAPOFF	Does not include any ECB heap storage if the ECB is running in a threaded environment.	

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZASER HELP
  ZASER ?
  ZASER SNAP HELP
  ZASER SNAP ?
- The system error options in effect for an initial program load (IPL) from the loader general file (LGF) are initialized using the LGF version of keypoint A, and are distinct from those in effect on an IPL from the prime module. The LGF version of

keypoint A is not updated when you enter the ZASER command after an IPL from the LGF, and neither is the version on the prime module. As a result, ZASER commands entered after an IPL from the LGF are not preserved across the next IPL of the TPF system.

- There are no multiple database function (MDBF) subsystem-unique options.
- Enter the ZDSER command to display the system error options and the SNAP options.

Examples

In the following example, a request is made to log system error messages to tape, to the operator console, and to route OPR dumps to the real-time processor.

```
User:   ZASER LOG CONSON PRT ECB CNTRLD

System: ASER0000I 15.28.54 - OK
```

In the following example, a request is made to suppress duplicate dumps and to write SNAP messages and dump data to the operator console.

```
User:   ZASER SNAP NODUPL CONSALL

System: ASER0000I 15.28.54 - OK
```

Related Information

See *TPF General Macros* for more information about system error dumps (SERRC) and SNAP dumps (SNAPC).

ZATIM–Add Sysplex Timer Information to Keypoint I

Use the following command to add Sysplex Timer information to Keypoint I when time-of-day (TOD) synchronization compatibility (TSC) hardware is used in the TPF system.

Requirements and Restrictions

- This command is valid only for the basic subsystem (BSS).
- Sysplex Timer information should be included in Keypoint I only when TSC hardware is connected to the Sysplex Timer.
- TSC hardware is required in a loosely coupled complex when a Sysplex Timer is the synchronization source and both TOD RPQ central processing complexes (CPCs) and Sysplex Timer-capable CPCs are present. The TSC hardware permits TOD RPQ CPCs to receive synchronization pulses from a Sysplex Timer.

Format

```
►►──ZATIM── ──timebase── ──TSC── ──STRNM-name── ──NETID-netid──────────────────►

►─ ──STRID-strid── ──SSA-ssa──────────────────────────────────────────────────►◄
```

timebase
: is the time base specified in hours (*hh*) and minutes (*mm*).

  Note: You do ***not*** need to enter the ZDTIM command to determine the correct time base because the time base does not need to match. This variable is included to keep the format of the ZATIM commands consistent.

STRNM-*name*
: is a 1- to 4- character Sysplex Timer name.

NETID-*netid*
: is a Sysplex Timer network ID from 0–31.

STRID-*strid*
: is a Sysplex Timer ID from 0–15.

SSA-*ssa*
: is a synchronization selection address from 0–7. This is the TOD RPQ port that is connected to the TSC.

Additional Information

The IBM 9037 Sysplex Timer is part of the IBM Enterprise Systems Connection Architecture.

Examples

Sysplex Timer information for STR1 is added to keypoint I in the following example.

```
User:   ZATIM 1529 TSC STRNM-STR1 NETID-3 STRID-1 SSA-6

System: ATIM0082I 15.28.54 TOD SYNCHRONIZATION COMPATIBILITY (TSC) INFORMATION
                           ADDED TO CTKI
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Generation* for more information about the Sysplex Timer.

ZATIM–Alter Time

Use this command to:
- Change the system time-of-day (TOD) clock
- Change the subsystem local standard time (LST) clock
- Synchronize the TOD clock to a Sysplex Timer.

Requirements and Restrictions

- Enter the ZDTIM command to determine the time base before you enter this command.
- If you enter this command when the TPF system is above 1052 state, you must cycle the TPF system to 1052 state to complete the time adjustments.
- You can enter this command with the TOD parameter only in 1052 state.
- In a loosely coupled system, all other active processors must be in 1052 state in order to change the TOD clock without the BP option.
- The ZATIM command does not adjust time-initiated functions (that is, functions that were started by using the CRETC macro).

Format

```
►►──ZATIM── ──timebase── ──────────────────────────────────────────────────────►

►──┬──┬─SET──┬── ──hhmm──────────────────────────────────────────────────┬────►◄
   │  └─ADD──┘                                                            │
   ├─TOD── ──┬─hhmmss── ──┬─mmddyy───┬─┬──┬────────────┬──┬──────────────┬──┬──────┬─┤
   │         │            └─ddmmmyy──┘ │  └──ALTPORT───┘  └──STRNM-name──┘  └──BP──┘ │
   │         └─GOOD────────────────────┘                                          │
   ├─TOD── ──STR──┬────────────┬──┬──────┬──────────────────────────────────────┤
   │              └──ALTPORT───┘  └──BP──┘                                      │
   └─STR── ──GOOD───────────────────────────────────────────────────────────────┘
```

timebase
: is the time base specified in hours (*hh*) and minutes (*mm*).

  Note: Enter the ZDTIM command to determine the time base.

SET
: changes the subsystem LST clock.

  Note: You can specify this parameter only when cycling is enabled. Enter the ZPSMS command to enable cycling.

ADD
: adds the specified hours and minutes to the LST clock.

hhmmss
: is the hours, minutes, and seconds.

mmddyy
: is the month (*mm*), day (*dd*), and year (*yy*).

  Note: Years ending in 00 through 41 are considered to be in the 21st century. Therefore, you can specify dates from January 1, 1942 through December 31, 2041.

All the processors in a loosely-coupled complex have the same format of days, months, and years.

ddmmmyy
is the alternate date format with day (*dd*), month (*mmm*), and year (*yy*). Specify the month as one of the following:

JAN
specifies January

FEB
specifies February

MAR
specifies March

APR
specifies April

MAY
specifies May

JUN
specifies June

JUL
specifies July

AUG
specifies August

SEP
specifies September

OCT
specifies October

NOV
specifies November

DEC
specifies December.

TOD
changes the time-of-day clock for the specified date.

The TOD parameter is valid only for the basic subsystem (BSS). It changes all the subsystem clocks when the subsystem is cycled above 1052 state.

Note: Enter the ZDTIM TOD or ZDTIM STR command to determine the time base for this command.

GOOD
indicates that the current value of the TOD clock is acceptable and the value will not be changed.

If you specify this parameter with the STR parameter, the TOD clock is synchronized to the Sysplex Timer. This command is transparent to all other central processor complexes (CPCs) in the complex.

STR
specifies that the time source is the Sysplex Timer.

ALTPORT
specifies that the time source is the Sysplex Timer on the alternate port.

This parameter is valid only when each port of a CPC goes to a different Sysplex Timer and these Sysplex Timers are uncoupled.

Note: You cannot specify this parameter when running under Processor Resource/Systems Manager (shared PR/SM).

STRNM-*name*
indicates that TOD synchronization compatibility (TSC) hardware is being used on the CPC and the Sysplex Timer should be used as the synchronization source, where *strname* is the 1- to 4-character name of the Sysplex Timer.

Note: TSC hardware is required when there are TOD RPQ CPCs and Sysplex Timer CPCs in the same loosely coupled complex.

BP
makes the TOD clock for the current processor the master TOD clock in a loosely coupled complex.

Note: You can specify this parameter only in a loosely coupled complex.

Additional Information

- Enter **ZDTIM** (with no parameters) to display the value of a subsystem LST clock. Use this value as the base time if you change the subsystem LST clock.
- When you change a subsystem LST clock, the TPF system immediately changes the time difference between the system clock and the subsystem clocks, the subsystem Greenwich clocks, and the subsystem perpetual clocks. In addition, if the subsystem is above 1052 state, the global clocks and the global calendar fields (if appropriate) are also changed. If the subsystem is not above 1052 state, the global clocks and the global calendar are updated when the subsystem is cycled above 1052 state.
- When a subsystem is in 1052 state, the SET parameter uses the last midnight value of the subsystem perpetual clock to set the subsystem clock and to determine the time difference between the system clock and the subsystem clock.
- When a subsystem is in 1052 state, the ADD parameter calculates what the time would be if the subsystem was above 1052 state, and adds the time you specify to set the subsystem clock and the new time difference between the system clock and the subsystem clock.
- After the TOD clock is set in the current processor, the other active processors are notified that the TOD clock was changed. All loosely coupled processors in 1052 state are automatically resynchronized. You must perform an initial program load (IPL) on the other processors above 1052 state in order for them to update and synchronize their TOD clocks to the new time. If an attempt is made to cycle one of the other processors above 1052 state without having synchronized its TOD clock, the cycle is canceled.
- In a loosely coupled system, if the TPF system is waiting for TOD clock confirmation and there are other active processors above 1052 state, you must specify the BP parameter when you enter the ZATIM TOD command.
- When you enter the ZATIM command, loosely coupled processors above 1052 state are notified that they are no longer synchronized with the complex. You must perform an IPL to resynchronize the clocks, or you must cycle to 1052 state and enter the ZATIM command again. The processors in 1052 state are resynchronized automatically.

- If you enter **ZATIM TOD GOOD BP** from the master processor, all other processor clocks are ***not*** marked unconfirmed. These processors are ***not*** notified that the TOD value was changed.
- Enter **ZATIM STR GOOD** to synchronize a single CPC with a Sysplex Timer. You do not need to perform an IPL for the CPC. The CPC connected to the Sysplex Timer has its ports verified; if the Sysplex Timer ports connected to the CPC are operational and synchronized with the master synchronization source, the CPC is synchronized to the remote Sysplex Timer. This command is transparent to all other CPCs in the complex.
- When you change the system TOD clock, the TPF system immediately updates the system clock, the system Greenwich clock, and the system perpetual clocks. The time differences between the system clock and each of the subsystem clocks are not affected, and the subsystem clocks (and date, if appropriate) are changed when the subsystem is cycled above 1052 state.
- The IBM 9037 Sysplex Timer is part of the IBM Enterprise Systems Connection Architecture.

Examples

The basic subsystem (BSS) LST clock is changed to 11 a.m. in the following example. The time difference between the system clock and the BSS clock is updated as well.

```
User:   ZDTIM

System: DTIM0003I 11.26.32 SUBSYSTEM BSS  CLOCKS ARE NOT RUNNING

User:   ZATIM 1126 SET 1100

System: ATIM0001I 11.00.32 SUBSYSTEM BSS  LOCAL STANDARD TIME
```

In the following example, the TOD clock is set to the time and date that is given by the Sysplex Timer. (You can enter the ZDTIM STR command to determine the time and date of the Sysplex Timer.)

```
User:   ZDTIM TOD

System: CLKS0010I 11.00.32 TIME OF DAY CLOCK LOCAL STANDARD TIME
                           TIME: 14.56.54  DATE: 05/24/94
        CLKS0081I 11.00.32 SOURCE: MASTER  STATUS: CONFIRMED
                           MASTER- CPUID: B  SERIAL: 020410  MODEL: 3090

User:   ZATIM 1456 TOD STR

System: CLKS0010I 11.00.32 TIME OF DAY CLOCK LOCAL STANDARD TIME
                           TIME: 14.57.10  DATE: 05/24/94
        ATIM0002I 11.00.32 TIME OF DAY CLOCK SET
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Generation* for more information about the system clocks.

ZATME–Reply to Date Change Message

Use this command to continue or cancel cycling a subsystem above 1052 state.

Requirements and Restrictions

You can enter this command only when you are prompted by the TPF system.

Format

```
►►──ZATME── ──┬─GOOD─┬────────────────────────────────────────────►◄
              ├─CNCL─┤
              └─CONT─┘
```

GOOD
allows the day boundary to be crossed; that is, continues the cycle-up request.

CONT
allows the ZATIM function to continue even though all processors are not in 1052 state.

CNCL
does not allow the day boundary to be crossed (cancels the cycle-up request), or cancels the ZATIM function.

Additional Information

- You can enter the ZATIM command to change the subsystem clocks before you enter this command.
- If you cancel the cycle-up request, you can cycle the TPF system up at a later time.
- If you cancel the ZATIM function, you can start it again at a later time.

Examples

In the following example, the TPF system continues cycling up the basic subsystem (BSS).

```
System: CLKS0011A 08.40.31 THE BSS WILL CHANGE DATE WHEN CYCLED UP
        TO CONTINUE OR TO CANCEL THE CYCLE UP - ENTER A ZATME MESSAGE

User:   ZATME GOOD

System: ATME0001I 00.00.00 ZCYCL WILL CONTINUE AS REQUESTED
        CLKS0013I 13.31.29 THE BSS WILL CONTINUE TO CYCLE UP
```

Related Information

See *TPF Main Supervisor Reference* for more information about cycling the TPF system.

ZAURS ABRT–Abort Unit Record Job

Use this command to end the jobs on a unit record device.

Requirements and Restrictions

None.

Format

```
►►──ZAURS ABRT── ──┬─name────┬──────────────────────────────────────────►◄
                   ├─address─┤
                   └─ALL─────┘
```

name
is the name of a unit record device, which can be CR1–CR3 for card readers and PR1–PR7 for printers.

address
is the 3-digit hexadecimal address of a unit record device.

ALL
ends the jobs on all the unit record devices.

Additional Information

None.

Examples

The jobs on the PR1 printer are ended in the following example.

```
User:   ZAURS ABRT PR1

System: AURS0051I 12.12.47 ABORT REQUEST SUMMARY
         NOT ABORTED  NONE
         ABORT STARTED ON  PR1
         SUSPENDED AND ABORT STARTED  NONE
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAURS AURT–Alter Unit Record Status Table

Use this command to change the address and device type of a unit record device or to specify that you cannot use a particular unit record device.

Requirements and Restrictions

None.

Format

```
►►──ZAURS AURT── ──name── ──┬──┬─1403──┬── ──address──┬──────────────────────►◄
                            │  ├─2540R─┤              │
                            │  ├─3211──┤              │
                            │  └─3505──┘              │
                            └─INVAL───────────────────┘
```

name
: is the name of the unit record device that you want to change, which can be CR1–CR3 for card readers and PR1–PR7 for printers.

1403
: specifies that the unit record device is a 1403 printer.

2540R
: specifies that the unit record device is a 2540R card reader.

3211
: specifies that the unit record device is a 3211 printer.

3505
: specifies that the unit record device is a 3505 card reader.

address
: is a 3-digit hexadecimal device address.

INVAL
: specifies that the unit record device cannot be used. That is, the device address is marked as not valid.

Additional Information

None.

Examples

The following information is displayed in the examples:

NAME
: is the symbolic name of the device.

DEV
: is the device type.

ADR
: is the device address, or INV if you cannot use the device.

STATUS
: is the status of the device, which can be:

PC
: Stops on print checks

A
: Assigned

S
: Suspended

STATUS BYTES
: shows the status bytes for the device, which are defined in the UR1ST DSECT.

The address and device type of the PR1 printer are changed in the following example.

```
User:   ZAURS AURT PR1 1403 00E

System: AURS0050I 17.06.42 UNIT RECORD STATUS
         NAME DEV  ADR STATUS STATUS BYTES

          PR1 1403 00E        80000000
```

The PR2 printer can no longer be used in the following example. Notice that the address for the PR2 printer is marked as not valid (INV).

```
User:   ZAURS AURT PR2 INVAL

System: AURS0050I 00.17.46 UNIT RECORD STATUS
         NAME DEV  ADR STATUS STATUS BYTES

          PR2 1403 INV        80000000
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAURS DURT–Display Unit Record Status Table

Use this command to display information about a unit record device.

Requirements and Restrictions

None.

Format

```
                     ┌─,──────┐
                     ▼        │
►►──ZAURS DURT── ──┬─name─────┴┬──────────────────────────────────►◄
                   └─ALL───────┘
```

name
: is the name of a unit record device, which can be CR1–CR3 for card readers and PR1–PR7 for printers.

ALL
: displays information about all the unit record devices.

Additional Information

None.

Examples

The following information is displayed in the examples:

NAME
: is the symbolic name of the device.

DEV
: is the device type.

ADR
: is the device address, or INV if you cannot use the device.

STATUS
: is the status of the device, which can be:

  PC
  : Stops on print checks

  A
  : Assigned

  S
  : Suspended

STATUS BYTES
: shows the status bytes for the device, which are defined in the UR1ST DSECT.

Information about the PR1 and PR2 printers is displayed in the following example.

```
User:   ZAURS DURT PR1,PR2

System: AURS0050I 00.17.46 UNIT RECORD STATUS
         NAME DEV  ADR STATUS STATUS BYTES

          PR1 1403 00E A      80000000
          PR2 1403 INV        80000000
```

Information about all the unit record devices is displayed in the following example.

```
User:   ZAURS DURT ALL

System: AURS0050I 00.17.46 UNIT RECORD STATUS
         NAME DEV  ADR STATUS STATUS BYTES

          PR1 1403 00E        80000000
          PR2 1403 INV        80000000
          PR3 3211 INV        90000000
          PR4 1403 03E        80000000
          PR5 1403 04E        80000000
          CR1 3505 00C        50000000
          CR2 3505 00F        50000000
          CR3 2540 01F        40000000
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAURS FCBL–Load 3211 Form Control Buffer

Use this command to load a form control buffer to a 3211 unit record printer.

Requirements and Restrictions

None.

Format

```
►►──ZAURT FCBL── ──┬─name────┬── ──┬─SD1─┬────────────────────────────────►◄
                   └─address─┘     ├─SD2─┤
                                   ├─SD3─┤
                                   └─SPR─┘
```

name
is the name of a 3211 unit record printer, which can be PR1–PR7.

address
is the 3-digit hexadecimal address of a 3211 unit record printer.

SD1
loads the SD1 form control buffer.

SD2
loads the SD2 form control buffer.

SD3
loads the SD3 form control buffer.

SPR
loads the SPR form control buffer.

Additional Information

- The form control buffers are defined in the CUAG program.
- The 4248 Impact Line Printer, Model 2, can run in 3211 emulation mode.

Examples

The SD1 form control buffer is loaded to the printer at the specified address in the following example.

```
User:   ZAURS FCBL 01E SD1

System: AURS0013I 12.12.47 FCB LOADED- RESET FORM TO LINE 1
```

The SD1 form control buffer is loaded to the PR2 printer in the following example.

```
User:   ZAURS FCBL PR2 SD1

System: AURS0013I 10.18.59 FCB LOADED- RESET FORM TO LINE 1
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAURS LUCS–Load Universal Character Set Buffer

Use this command to load a universal character set to a unit record printer.

Requirements and Restrictions

None.

Format

```
►►──ZAURT LUCS── ──address── ──┬─A11─┬──┬──────┬──────────────────────────►◄
                               ├─G11─┤  └─ ──U─┘
                               ├─H11─┤
                               ├─P11─┤
                               ├─T11─┤
                               ├─HP──┤
                               ├─HN──┤
                               ├─PN──┤
                               ├─QN──┤
                               └─TN──┘
```

address
: is the 3-digit hexadecimal address of a unit record printer.

A11
: loads the standard commercial (dualed) universal character set for a 3211 printer.

G11
: loads the ASCII universal character set for a 3211 printer.

H11
: loads the standard scientific (dualed) universal character set for a 3211 printer.

P11
: loads the PL 1 universal character set for a 3211 printer.

T11
: loads the text printing universal character set for a 3211 printer.

HP
: loads the AN/HN preferred universal character set for a 1403 printer.

HN
: loads the AN/HN standard universal character set for a 1403 printer.

PN
: loads the PN standard universal character set for a 1403 printer.

QN
: loads the QN standard universal character set for a 1403 printer.

TN
: loads the TN text printing universal character set for a 1403 printer.

U allows data checks to be performed.

Additional Information

- The 3211 universal character sets are defined in the CUAI program. The 1403 universal character sets are defined in the CUAH program.
- The 4248 Impact Line Printer, Model 2, can run in 3211 emulation mode.

Examples

The HP universal character set character set is loaded to the specified unit record printer in the following example. Data checks are allowed.

```
User:   ZAURS LUCS 00E HP U

System: AURS0015I 17.15.27 UCSB LOADED
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAURS UTIL–Unit Record Utility Program

Use this command to do the following:

- Read data from the first available unit record card reader and write it to the first available unit record printer.
- Read data from the first available unit record card reader and write it to a UTL tape.
- Read data from a UTL tape and write it to the first available unit record printer.

Requirements and Restrictions

Mount the UTL tape, if necessary.

Format

```
►►──ZAURS UTIL── ──┬─CP─┬──────────────────────────────────────────────────►◄
                   ├─CT─┤
                   └─TP─┘
```

CP
: reads data from the first available card reader and writes it to the first available printer.

CT
: reads data from the first available card reader and writes it to a UTL tape.

TP
: reads data from a UTL tape and writes it to the first available printer.

Additional Information

None.

Examples

Data is read from the first available unit record card reader and written to the first available unit record printer in the following example.

```
User:   ZAURS UTIL CT

System: AURS0052I 00.17.46 OPT CP COMPLETED
```

Related Information

See *TPF Data Communications Services Reference* for more information about unit record support.

ZAUTH–Display or Alter Display Device Logon Authority

Use this command to:

- Display or change the LOGI logon authority for a display device
- Initialize the records in the display device application authorization table (TAPP).

Requirements and Restrictions

None.

Format

```
►►──ZAUTH── ──┬─Display── ──LIT-lniata──────────────────────────┬──────────►◄
              ├─ALTer── ──LIT-lniata── ──┬─ALL─────────────────┤
              │                          ├─PLOG-appl───────────┤
              │                          │          ┌─,────┐   │
              │                          │          ▼      │   │
              │                          └──┬─A─┬─-───appl─┴───┤
              │                             ├─D─┤              │
              │                             └─R─┘              │
              └─INITiali──────────────────────────────────────┘
```

Display
displays the LOGI logon authority for the specified display device.

Alter
changes the LOGI logon authority for the specified display device.

lniata
is the 6-digit hexadecimal line address, interchange address, and terminal address (LNIATA) of a display device.

ALL
authorizes a display device to log on to all applications using the LOGI command.

PLOG
permanently logs on a display device to the specified application.

A authorizes a display device to log on to additional applications using the LOGI command.

R authorizes a display device to log on to the specified applications using the LOGI command. This authorization replaces any previous authorization.

D deletes LOGI logon authority for the specified applications.

appl
is a 1- to 4-character alphanumeric application name.

INITiali
initializes the TAPP records for all display devices.

Additional Information

None.

Examples

Information about the logon authority of the specified display device is displayed in the following example.

```
User:   ZAUTH DISP LIT-4E0000

System: CLGH0022I 09.11.21
        LIT-4E0000B
        APPL-GMT5
        LOG AUTH-ALL
```

The logon authority of the specified display device is changed in the following example. The display device is now permanently logged on to the specified application.

```
User:   ZAUTH ALTER LIT-4E0000 PLOG-GMT5

System: CLGG0022I 09.12.32
        LIT-4E0000B
        ALTER COMPLETE
        AUTH WAS ALL
```

Related Information

See *TPF Data Communications Services Reference* for more information about the log processor for non-SNA display devices.

ZAVSN–Alter Volume Serial Number

Use this command to change the volume serial number of an offline DASD.

An *offline DASD* is one that is physically mounted and in ready state, but is not already mounted for use by the TPF system.

Requirements and Restrictions

The DASD must be offline before you enter this command.

Format

```
►►──ZAVSN── ──ccud── ──volume──────────────────────────────────────────────►◄
```

ccud
is the 3- to 4-digit hexadecimal hardware address of a DASD.

volume
is a 6-character alphanumeric volume serial number for the DASD.

Note: You can specify special characters if you begin and end the volume serial number with the ' character. However, the volume serial number can contain only 4 characters.

Additional Information

After changing the volume serial number, the DASD remains offline to the TPF system. To place the DASD online, use one of the following commands:

- ZAMOD, for real-time DASD
- ZFMNT, for general file DASD
- ZDSMG, for general data set DASD.

Examples

The volume serial number is changed for the specified hardware address in the following example.

```
User:   ZAVSN 4E7 BP0998

System: AVSN0000I 15.27.09 VSN ON DISK 04E7 IS BP0999
        AVSN0001I 15.27.09 ALTER OF VSN COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZBOLT–Start a BSC Online Test

Use this command to send a request-for-test message from a binary synchronous communication (BSC) point-to-point or tributary station to:

- Another system
- An online test executive program (OLTEP) system
- An online test stand alone executive program (OLTSEP) system
- Any system that follows the OLTEP or OLTSEP line protocol.

Requirements and Restrictions

None.

Format

```
►►──ZBOLT── ──BS── ──line── ──┬─00─┬── ──repeat──────────────────────────►
                              ├─01─┤
                              ├─02─┤
                              ├─04─┤
                              ├─05─┤
                              ├─06─┤
                              ├─14─┤
                              ├─15─┤
                              ├─16─┤
                              ├─19─┤
                              ├─20─┤
                              ├─21─┤
                              └─22─┘

►──┬──────────────────────────────┬──────────────────────────────────────►◄
   │    (1)                       │
   └─ ───────┬─C─┬── ──message────┘
             └─X─┘
```

Notes:

1 You must specify these parameters only when you specify the 00 or 01 parameter.

line
is a 2-digit hexadecimal symbolic line number.

00 receives and acknowledges the specified messages.

Note: The messages consist of data plus line control.

01 transmits the test message that was received in the RFT text field.

02 transmits an EBCDIC full transparent message:

DLE STX (256 8-bit bytes, 0–255) DLE ETX

04 transmits an EBCDIC normal text message:

STX SYN SYN (245 8-bit bytes, excluding control characters) ETX

05 transmits a USASCII normal text message:

STX SYN SYN (117 7-bit bytes, excluding control characters) ETX

06 transmits a USASCII alphanumeric text message:

STX SYN SYN (36 bytes, A–Z, 0–9) ETX

14 transmits an EBCDIC alphanumeric message:

STX SYN SYN (36 bytes, A–Z, 0–9) ETX

15 transmits an EBCDIC all-zeros weak pattern message:

STX SYS SYN (74 bytes X'00' and 6 bytes SYN) ETX

16 transmits an EBCDIC alternate zero-on weak pattern message:

STX SYN SYN (40 bytes X'AA', 40 bytes X'55') ETX

19 transmits an EBCDIC weak pattern transparency message:

DLE STX (280 bytes X'00', 10 bytes SYN) DLE ETX

20 transmits an EBCDIC 80-byte transparent text message:

DLE STX (U–Z, 0–9, X'00'–X'3F') DLE ETX

21 transmits an EBCDIC 120-byte transparent text message:

DLE STX (A–Z, 0–9, X'00'–X'53') DLE ETX

22 transmits an EBCDIC 144-byte transparent text message:

DLE STX (A–Z, 0–9, X'00'–X'6B') DLE ETX

repeat
is the number of times the message is sent from 1–99.

C specifies that the message contains alphanumeric characters. Character data is sent in nontransparent mode exactly as it is keyed on the CRAS, including new-line characters.

X specifies that the message contains hexadecimal characters. Hexadecimal data is converted to the appropriate internal form and sent in transparent mode.

message
is the 0- to 290-byte message. If you specify hexadecimal characters, the message must contain an even number of bytes. New-line characters that are inserted between the hexadecimal digits are ignored.

Additional Information

You can enter the ZLASL command to prevent normal traffic from being queued on a BSC line while running online tests. To do this, set the send limit of the line to 0. Any traffic that is already queued is sent. Additional messages are not queued until the send limit (except online tests) is reset to a nonzero value.

Examples

In the following example, a USASCII normal text message is sent 3 times on line 46.

```
User:   ZBOLT BS 46 05 03

System: 18.10.09 BOLT- TEST 05 COUNT 03 IN RFT SENT ON LINE BS 46
```

In the following example, the ABC123 characters are received and acknowledged 99 times on line 46.

```
User:   ZBOLT BS 46 00 99 C ABC123

System: 18.11.51 BOLT- TEST 00 COUNT 99 IN RFT SENT ON LINE BS 46
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about BSC lines.

ZBROW ALTER–Alter the Contents or Access Mode of a Specified Collection

Use this command to change the contents or access mode of a collection.

With this command you can:

- Add new elements to a collection in the data store
- Modify elements in a collection with the qualified data
- Delete elements from a collection
- Change the access mode of a collection in the data store.

Requirements and Restrictions

- You must initialize TPF collection support (TPFCS) and define the target data store before you enter this command.
- The TPF system must have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.
- You must enter the ZBROW QUALIFY command with the appropriate parameters specified to qualify the target element in a collection before you enter this command with the ELEMENT parameter specified. The qualification parameters will be based on the collection type being accessed. See the collection support section in *TPF Database Reference* for more information about collection types.

Format

```
►►──ZBROW ALTer── ──┬─ELEment── ──colname── ──┬─ADD─────┬──┬──────────────►◄
                    │                         ├─MODify──┤  │
                    │                         └─DELete──┘  │
                    └─MODE── ──colname── ──┬─NORMAL─────┬──┘
                                           ├─NOCHANGE───┤
                                           ├─NODELETE───┤
                                           └─READONLY───┘
```

ELEment
: causes the contents of the specified collection to be changed.

colname
: is the 1- to 32-character alphanumeric name of the collection you want to change.

ADD
: adds the qualified element to the specified collection.

MODify
: updates the qualified element in the specified collection.

DELete
: deletes the qualified element from the specified collection.

MODE
: causes the access mode of the specified collection to be changed.

NORMAL
sets the access mode of the collection to NORMAL. All functions are allowed for the collection.

NOCHANGE
sets the access mode of the collection to NOCHANGE. The collection can only be read or deleted; it cannot be updated or added to.

NODELETE
sets the access mode of the collection to NODELETE. The collection can be read, updated, or added to.

Note: The mode applies to the collection and not to its elements so that an application can delete all the elements in a collection with this mode, but it cannot delete the actual collection itself.

READONLY
sets the access mode of the collection to READONLY. The collection can only be read; it cannot be updated, added to, or deleted.

Additional Information

- To display the current access mode of the collection, use the ZBROW DISPLAY command specifying the COLLECTION parameter.
- To display the current qualification, use the ZBROW QUALIFY command specifying the DISPLAY parameter.

Examples

In some of the following examples, the ZBROW QUALIFY command is shown only to make the examples more complete.

The recoup index collection access mode is changed to NORMAL in the following example.

```
User:    ZBROW QUAL SET DS-BANK1_DS

System:  BROW0607I 10.21.26 QUALIFICATION PROCESSING COMPLETED

User:    ZBROW ALT MODE DS_RECOUP NORMAL

System:  BROW0602I 10.23.40 BROWSER QUALIFIED FOR DSNAME BANK1_DS
System:  BROW0425I 10.23.40 ACCESS MODE FOR COLLECTION CHANGED
System:  BROW0410I 10.23.40 BROWSE OF COLLECTION COMPLETED
```

The qualified element of collection ARRAYL1 is modified in the following example.

```
User:    ZBROW QUAL SET DATA-1234567891011

System:  CSMP0097I 15.09.25 CPU-B SS-BSS  SSU-HPN  IS-01
         BROW0607I 15.09.25 QUALIFICATION PROCESSING COMPLETED

User:    ZBROW ALTER ELEMENT ARRAYL1 MODIFY

System:  CSMP0097I 15.09.25 CPU-B SS-BSS  SSU-HPN  IS-01
         BROW1120I 15.09.25 BEGIN DISPLAY OF THE QUALIFIED ELEMENT
         ABCDEFGHIJKLM
         ALTERED TO -
         1234567891011
         BROW1110I 15.09.25 ALTER REQUEST COMPLETED
```

The following example shows the addition of an element to an array collection.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 16.47.54 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 16.47.54 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - TESTDATA
        DATA LENGTH  - 30
        DATA PAD     - X%F0
        BROW0607I 16.47.54 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW ALTER ELEMENT ARRAYL1 ADD

System: CSMP0097I 16.50.22 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW1110I 16.50.22 ALTER REQUEST COMPLETED+

User:   ZBROW DISPLAY ELEMENT ARRAYL1 INDEX-201 RSA-0.40

System: CSMP0097I 16.55.14 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 16.55.14 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 16.55.14 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000001 LGH 00000030
         00000000- E3C5E2E3 C4C1E3C1 F0F0F0F0 F0F0F0F0 TESTDATA 00000000
         00000010- F0F0F0F0 F0F0F0F0 F0F0F0F0 F0F0F0F0 00000000 00000000
         00000020- F0F0F0F0 F0F0F0F0 F0F0F0F0 F0F0F0F0 00000000 00000000
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED+
        CSMP0097I 16.55.14 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0410I 16.55.14 BROWSE OF COLLECTION COMPLETED+
```

The following example shows the modification of the element at index 201.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 16.59.22 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 16.59.22 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - X%E7E7E7E7
        INDEX        - 201
        RSA          - 4
        BROW0607I 16.59.22 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW ALTER ELEMENT ARRAYL1 MOD

System: CSMP0097I 16.59.53 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW1120I 16.59.53 BEGIN DISPLAY OF THE QUALIFIED ELEMENT
        DATA
        ALTERED TO -
        XXXX
        BROW1110I 16.59.53 ALTER REQUEST COMPLETED+

User:   ZBROW DISPLAY ELEMENT ARRAYL1 INDEX-201 RSA-0.40

System: CSMP0097I 17.00.24 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 17.00.24 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 17.00.24 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000002 LGH 00000030
         00000000- E3C5E2E3 E7E7E7E7 F0F0F0F0 F0F0F0F0 TESTXXXX 00000000
         00000010- F0F0F0F0 F0F0F0F0 F0F0F0F0 F0F0F0F0 00000000 00000000
         00000020- F0F0F0F0 F0F0F0F0 F0F0F0F0 F0F0F0F0 00000000 00000000
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED+
        CSMP0097I 17.00.24 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0410I 17.00.24 BROWSE OF COLLECTION COMPLETED+
```

The following example attempts to delete the element at index 201 (the last element of the array). Note that the ZBROW DISPLAY command does not display the deleted element.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 17.02.14 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 17.02.14 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        INDEX        - 201
        BROW0607I 17.02.14 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW ALTER ELEMENT ARRAYL1 DEL

System: CSMP0097I 17.02.55 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW1110I 17.02.55 ALTER REQUEST COMPLETED+

User:   ZBROW DISPLAY ELEMENT ARRAYL1 INDEX-201 RSA-0.40

System: CSMP0097I 17.03.21 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 17.03.21 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 17.03.21 COLLECTION ELEMENT DISPLAY
        CLASS ID 00000000 SEQ CTR 00000000 LGH 00000000
        DATA LENGTH LESS THAN SPECIFIED RSA
        BROW0410I 17.03.21 BROWSE OF COLLECTION COMPLETED+
```

The following example shows the addition of an element to a keyset collection.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 17.06.24 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 17.06.24 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - SYSTEMTESTDATA1234
        RSA          - 10
        SEARCH       - KEY1
        BROW0607I 17.06.24 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW ALTER ELEMENT KEYSETS1 ADD

System: CSMP0097I 17.11.49 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW1110I 17.11.49 ALTER REQUEST COMPLETED+

User:   ZBROW QUAL DISP ALL

System: CSMP0097I 17.13.09 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 17.13.09 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - SYSTEMTESTDATA1234
        DATA LENGTH  - 40
        RSA          - 0
        SEARCH       - KEY1
        BROW0607I 17.13.09 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW DISPLAY ELEMENT KEYSETS1 USEQUAL

System: CSMP0097I 17.12.44 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 17.12.44 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 17.12.44 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000000 LGH 00000012
         00000000- E2E8E2E3 C5D4E3C5 E2E3C4C1 E3C1F1F2 SYSTEMTE STDATA12
         00000010- F3F4                                34
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED+
        CSMP0097I 17.12.44 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0410I 17.12.44 BROWSE OF COLLECTION COMPLETED+
```

The following example attempts to delete an element from a log collection.

```
User:   ZBROW ALTER ELEMENT LOGL1 DEL

System: CSMP0097I 17.15.13 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW1157E 17.15.13 DELETION OF LOG/KEYEDLOG ELEMENTS NOT ALLOWED
        BROW1151E 17.15.13 ALTER REQUEST FAILED+
```

Related Information

- See “ZBROW DISPLAY–Display Information about a Collection or Its Contents” on page 131 for more information about the ZBROW DISPLAY command.
- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW CLASS–Display Class Name Information

Use this command to display information about a class for TPF collection support (TPFCS).

Requirements and Restrictions

You ***must*** initialize TPFCS and define the target data store (DS) before you enter this command.

Format

```
►►──ZBROW CLAss── ──────────────────────────────────────────────────────────►

►──┬─ATTributes── ──Name-classname──────────────────────────┬──────────────►◄
   │           ┌─ ──ALL──────────┐                        │
   ├─DISplay───┼──────────────────┼────────────────────────────┤
   │           └─ ──Name-classname─┘  └─ ──SORTA─┘          │
   ├─DOCument── ──┬─Name-classname──┬──────────────────┬─┬─┤
   │              │                ├─ ──ALL───────────┤ │ │
   │              │                └─ ──Method-methodname─┘ │ │
   │              └─ALL──────────────────────────────────┘ │
   ├─INFO── ──Name-classname───────────────────────────────┤
   ├─LOCate── ──Name-classname── ──Method-methodname───────┤
   ├─MAP───────────────────────────────────────────────────┤
   ├─METhod── ──Name-classname──┬─────────────┬────────────┤
   │                            └─ ──SORTA─┘             │
   └─TREE── ──Name-classname────────────────────────────────┘
```

ATTributes
displays the names and values of all the class attributes of the class.

Name-*classname*
specifies the name of the class, where *classname* is a 1- to 64-character class name.

DISplay
displays the location of the specified class name or displays all the class names and their locations.

ALL
specifies one of the following:

- When you specify the DISPLAY parameter, the ALL parameter displays all class names and their locations.
- When you specify the DOCUMENT parameter, the ALL parameter dumps either of the following to the printer:
  - All the documentation for all methods for a class
  - All the documentation for all the classes and all methods in TPF collection support.

SORTA
sorts the class or method names displayed by location (address). If you do not

specify the SORTA parameter, names are displayed in no particular order. When used with the DISPLAY parameter, the SORTA parameter is only useful when ALL is specified.

DOCument
dumps documentation contained in the prologs of the specified class or all classes and its methods to the printer. The information that is dumped is based on the other parameters you specify, as follows:

- If you specify the NAME parameter, class information is dumped to the printer.
- If you specify the ALL parameter, all method information for a class is dumped to the printer.
- If you specify the METHOD parameter method information for a specified method and class is dumped to the printer.
- If you specify the ALL parameter without the NAME parameter information for all methods for all classes is dumped to the printer.

Method-*methodname*
specifies the name of the method, where *methodname* is a 1- to 64-character method name.

- When you specify this parameter with the DOCUMENT parameter, method information for the specified class and method is dumped to the printer.
- When you specify this parameter with the LOCATE parameter, the locations of all methods that start with the specified character string for the specified class are displayed.

Note: If no methods start with the specified character string for the specified class, the method location display table will be empty.

INFO
displays information about a specific class.

LOCate
displays the location of the specified method or methods for the specified class name.

MAP
dumps the location of all class names and their methods sorted by address to the printer.

METhod
displays the location, type, and name of all the methods for the specified class name. Type can be one of the following:

C Class

I Instance

P Private

TREE
displays the names of all classes from which the specified class inherits.

Note: The class inheritance for class dictionary is displayed in reverse order; that is, the class most directly inherited from is displayed first.

Additional Information

None.

Examples

The class attributes for a TPFCS class are displayed in the following example. This example shows that the method trace table is active (01) and dump creation on a `TO2_getErrorText` function call is not active (00). See “ZOODB SET–Set TPF Collection Support” on page 1102 for more information about setting the class attributes.

```
User:   ZBROW CLASS ATTR NAME-TO2

System: BROW0206I 17.31.04 CLASS ATTRIBUTES DISPLAY
        ATTRIBUTE                          VALUE
        TO2_C_ASSEM_DATE                   08/28/96
        TO2_C_ASSEM_TIME                   08.22
        TO2_C_USER_TRACE_COUNT             00000020
        TO2_C_METHOD_COUNT                 00000200
        TO2_C_INITIALIZED                  01
        TO2_C_METHOD_TRACE                 01
        TO2_C_DUMP_GET_TEXT                00
        END OF DISPLAY
        BROW0210I 17.31.04 BROWSE OF CLASS COMPLETED
```

The location of a class dictionary is shown in the following example.

```
User:   ZBROW CLASS DISP NAME-DICTIONARY

System: BROW0201I 13.34.04 LOCATION - 01DF30A8  NAME - DICTIONARY
        BROW0210I 13.34.04 BROWSE OF CLASS COMPLETED
```

All class names are displayed in the following example.

```
User:   ZBROW CLASS DISP ALL

System: BROW0202I 13.24.10 CLASS LOCATION DISPLAY
        LOCATION    ID1     ID2     NAME
        013A14A0       1    0004    COLLECTION
        013D73C0       2    0008    Collect
        013DBA78       3    000C    ARRAY
        013E3FA0       4    0010    Set
        013E55B8       5    0014    Ordered
        013DE1D0       6    0018    Log
        013E8500       7    001C    Dictionary
        014513C8       8    0020    TPF_Dictionary
        013B3220       9    0024    ENV
        0144B020      10    0028    DB_OBJ
        0145E5E0      11    002C    xternalObject
        013C0E40      12    0030    USER_OBJ
        013BB7E8      13    0034    PIDentry
        013DEE10      14    0038    KeyedLog
        013E11A0      15    003C    Bag
        013E2760      16    0040    ByteArray
        0139D7A0      17    0044    TPFservice
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

All the class names and locations (in ascending address sequence) are displayed in the following example.

```
User:   ZBROW CLASS DISP ALL SORTA

System: BROW0202I 13.25.29 CLASS LOCATION DISPLAY
        LOCATION    ID1     ID2     NAME
        0139D7A0     17    0044    TPFservice
        013A0650    120    01E0    TO2_HEAP
        013A14A0      1    0004    COLLECTION
        013A5288    188    02F0    ObjectPart
        013A69A0    119    01DC    TO2
        013AEE18    196    0310    TO2_CLASS_ENTRY
        013AF360    172    02B0    TaskServices
        013AFCE8    173    02B4    TaskServiceClass
        013B0510    174    02B8    TaskServiceObject
        013B0FA8     26    0068    ObjectTable
        013B3220      9    0024    ENV
        013B3DC0     25    0064    PIDnumber
        013B5BA8     49    00C4    PIDnumberPersistent
        013B7A78    168    02A0    PIDnumberPersistentFormat0
        013B8478    169    02A4    PIDnumberPersistentFormat1
        013B8C30    170    02A8    PIDnumberPersistentFormat2
        013B9B58     50    00C8    PIDnumberTemporary
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

The class documentation for the class DICTIONARY is dumped to the printer in the following example.

```
User:   ZBROW CLASS DOC NAME-DICTIONARY

System: BROW0210I 08.28.07 BROWSE OF CLASS COMPLETED
```

The class documentation for the NEW method of class DICTIONARY is dumped in the following example.

```
User:   ZBROW CLASS DOC NAME-DICTIONARY METHOD-NEW

System: BROW0210I 08.28.07 BROWSE OF CLASS COMPLETED
```

The class documentation for all of the methods of class DICTIONARY are dumped in the following example.

```
User:   ZBROW CLASS DOC NAME-DICTIONARY ALL

System: BROW0210I 08.28.07 BROWSE OF CLASS COMPLETED
```

The documentation for all TPFCS collections is dumped in the following example.

```
User:   ZBROW CLASS DOC ALL

System: BROW0210I 08.28.07 BROWSE OF CLASS COMPLETED
```

A sample of printer output is shown for ZBROW CLASS DOCUMENT METHOD NAME in the following example.

```
*********************************************************************************
 ** Method Name - ADDALLFROM

      This method is invoked to add all the elements of a source
    collection to the specified object collection. No validation
    is done on the source collection to determine if it can be
    added to the target collection.

    Input  R1  ->  collection to add to target collection
    Output R15 ->  target collection
                   TO2_ERROR_EMPTY  source collection empty
```

A sample of printer output is shown for ZBROW CLASS DOCUMENT NAME ALL in the following example.

```
*********************************************************************************
*********************************************************************************
 ** Method Name - deleteCollection


    Input  R2   ->   address of structure object to delete
    Output R15  ->   none


*********************************************************************************
 ** Method Name - chainAllocatedFA


    Input  R1   -> word1 ->  core block for new record
                   word2 ->  directory entry (new record)
                   word3 ->  RecID and RCC values
           R2   ->   address of structure object
    Output R15  ->   core block address if successful
                     logic error code if directory contains 0 FA.
```

The class information for the OIDentry class is displayed in the following example.

```
User:   ZBROW CLASS INFO NAME-OIDENTRY

System: BROW0205I 18.42.48 CLASS INFORMATION DISPLAY
        NAME - OIDentry
        ID - x0000000D , x00000034
        ADDR - x01CA6EA8 LGH - x00004B8A
        CLASS METHOD COUNT - x00000001 INSTANCE METHOD COUNT - x00000029
        HASH VALUE - x00FB SYNONYM COUNT - x0012
        END OF DISPLAY
        BROW0210I 18.42.48 BROWSE OF CLASS COMPLETED
```

The location of all methods that start with `at` for class DICTIONARY are displayed in the following example.

```
User:   ZBROW CLASS LOCATE NAME-DICTIONARY METHOD-at

System: BROW0203I 13.34.12 METHOD LOCATION DISPLAY
        ADDRESS     TYPE    NAME
        01DF3CA4    I       atKey
        01DF4344    I       atKeyWithBuffer
        01DF3E08    I       atKeyPut
        01DF408C    I       atNewKeyPut
        01DF42B8    I       at
        01DF42B8    I       atPut
        END OF DISPLAY
        BROW0210I 13.34.12 BROWSE OF CLASS COMPLETED
```

The information for all classes defined in TPF collection support is dumped to the printer in the following example.

```
User:   ZBROW CLASS MAP

System: BROW0210I 13.34.12 BROWSE OF CLASS COMPLETED
```

A sample of printer output from ZBROW CLASS MAP is shown in the following example.

```
NAME - TPFservice
ID - x00000011 , x00000044
ADDR - x01C84D70 LGH - x0000357C
CLASS METHOD COUNT - x0000002F INSTANCE METHOD COUNT - x00000000
HASH VALUE - x0000 SYNONYM COUNT - x0000
ADDR 01C8534C TYPE C NAME inheritFixedMethods
ADDR 01C85658 TYPE C NAME getStorage
ADDR 01C856F8 TYPE C NAME freeStorage
ADDR 01C85798 TYPE C NAME releaseBlock
ADDR 01C8584C TYPE C NAME getBlock
ADDR 01C858FC TYPE C NAME getSSindex
ADDR 01C859B8 TYPE C NAME getFileAddress
ADDR 01C85A9C TYPE C NAME readRecord
ADDR 01C85BB4 TYPE C NAME readRecordNoWait
ADDR 01C85C58 TYPE C NAME lockReadRecord
ADDR 01C85D7C TYPE C NAME fileUnlockRecord
ADDR 01C85E68 TYPE C NAME fileRecord
ADDR 01C85F88 TYPE C NAME fileRecordNoWait
ADDR 01C86030 TYPE C NAME fileRecordWithReleaseNoWait
ADDR 01C860E0 TYPE C NAME fileUnlockRecordWithReleaseNoWait
ADDR 01C861A0 TYPE C NAME unLockRecord
```

The location and type (Class, Internal, or Private) of all methods for class DICTIONARY are displayed in the following example.

```
User:   ZBROW CLASS METHOD NAME-DICTIONARY

System: BROW0203I 13.34.12 METHOD LOCATION DISPLAY
        ADDRESS     TYPE    NAME
        013E8F9C    C       createUser
        013E8D34    C       newX
        013E898C    C       new
        013E8B60    C       newFix
        013E90E4    I       atKey
        013E9784    I       atKeyWithBuffer
        013E9248    I       atKeyPut
        013E94CC    I       atNewKeyPut
        013E960C    I       removeKey
        013E99E0    I       newCursorRead
        013E9AD8    I       newCursorWrite
        013E9BD0    I       newCursorReadWrite
        013E96F8    I       at
        013E96F8    I       atPut
        013E96F8    I       add
        013E96F8    I       remove
        013E988C    I       convertToDASD
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

All the class methods and locations (in ascending address sequence) are displayed for class DICTIONARY in the following example.

```
User:   ZBROW CLASS METHOD NAME-DICTIONARY SORTA

System: BROW0203I 13.41.34 METHOD LOCATION DISPLAY
        ADDRESS     TYPE    NAME
        013E898C    C       new
        013E8B60    C       newFix
        013E8D34    C       newX
        013E8F9C    C       createUser
        013E90E4    I       atKey
        013E9248    I       atKeyPut
        013E94CC    I       atNewKeyPut
        013E960C    I       removeKey
        013E96F8    I       at
        013E96F8    I       atPut
        013E96F8    I       add
        013E96F8    I       remove
        013E9784    I       atKeyWithBuffer
        013E988C    I       convertToDASD
        013E99E0    I       newCursorRead
        013E9AD8    I       newCursorWrite
        013E9BD0    I       newCursorReadWrite
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

The class inheritance tree for class DICTIONARY is displayed in the following example.

```
User:   ZBROW CLASS TREE NAME-DICTIONARY

System: BROW0204I 13.34.12 CLASS INHERITANCE DISPLAY
        NAME Dictionary
        NAME CollectKey
        NAME Collect
        NAME ObjectPart
        NAME OBJECT
        END OF DISPLAY
        BROW0210I 13.34.12 BROWSE OF CLASS COMPLETED
```

Related Information

See the following books for more information about TPFCS:

- *TPF Application Programming*
- *TPF Concepts and Structures*
- *TPF Database Reference*.

ZBROW COLLECTION–Perform Maintenance on a Collection

Use this command to display information about the specified collection and to perform maintenance on the specified collection.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store (DS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZBROW COLlection── ──────────────────────────────────────────────────────►

►──┬─ADDAll── ──targetname── ──sourcename──────────────┬──────────────────►◄
   ├─ATTributes── ──colname────────────────────────────┤
   ├─CAPTure── ──colname───────────────────────────────┤
   ├─COPY── ──colname── ──newname──────────────────────┤
   ├─COPYDD── ──colname── ──newname── ──ddname─────────┤
   ├─DELete── ──colname────────────────────────────────┤
   ├─DUMP── ──colname──┬─────────────────────────┬─────┤
   │                   └─ ──KEYPath-keypathname──┘     │
   ├─EMPTY── ──colname─────────────────────────────────┤
   ├─MIGrate── ──colname── ──newname───────────────────┤
   ├─NAMe── ──colname──────────────────────────────────┤
   ├─PARTs── ──colname─────────────────────────────────┤
   ├─RECLaim── ──colname───────────────────────────────┤
   ├─RECONstruct── ──colname── ──┬─DATA──────┬─────────┤
   │                             ├─KEYS──────┤         │
   │                             └─DIRECTory─┘         │
   ├─RESTore── ──colname── ──origname── ──newname──┤ Options 1 ├──┤
   └─VALidate── ──colname──┬─────────────────────────┬─┘
                           └─ ──KEYPath-keypathname──┘
```

Options 1:

```
├──┬──────────────────┬──┬────────────┬──────────────────────────────────┤
   └─ ──DDname-ddname─┘  └─ ──SHADOW──┘
```

ADDAll

adds all the elements of a source collection to the specified target collection in the most correct way.

Notes:

1. You cannot add elements from a collection that does not have keys to a collection that has keys.
2. The size of the largest element in the source collection must be less than or equal to the maximum element size in the target collection.

3. The size of the largest key in the source collection must be less than or equal to the maximum key size in the target collection.
4. When the source collection is a binary large object (BLOB), the target collection must be a BLOB.

targetname
is the name of the target collection. The target collection must previously exist.

sourcename
is the name of the source collection.

ATTributes
displays the names and values of all of the attributes of the internal components of the specified collection.

colname
is the assigned name of the collection.

CAPTure
specifies that the collection and any assigned properties will be written to an external device. Collections cannot be captured or restored using blocked tapes.

COPY
makes a copy of the specified source collection, including the following:

- Access mode
- Associated recoup index
- Dirty-read (DR) protect mode
- Key paths
- Properties
- Read-only attribute
- User class ID.

newname
is the new name of the collection that was copied, migrated, or restored. The new name cannot already be associated with a collection.

COPYDD
makes a copy of the specified source collection using the specified data definition for the copy.

ddname
is the data definition (DD) to apply to the new collection.

DELete
marks a collection for deletion. The actual deletion will be delayed until a later time. The collection name is not deleted. Use the ZBROW NAME command with the REMOVE parameter specified to remove the name.

DUMP
dumps the contents of the specified collection to the printer. If no key path is specified, the primary key path is used.

KEYPath-*keypathname*
specifies one of the following, where *keypathname* is the 1- to 16-character alphanumeric name of the key path:

- When you specify the DUMP parameter, this command dumps the collection by using the specified key path.

- When you specify the VALIDATE parameter, this command verifies the specified key path structure. To verify the primary key path of a collection, specify TO2_PRIME_KEYPATH as the key path name.

EMPTY
deletes all elements in the collection. If the collection specified is DS_DELETED and the data store is set to use delayed deletes, all pending deletes will be processed for the qualified data store.

MIGrate
migrates a specified source collection to a collection using new pools in the current format. This includes the following:

- Access mode
- Associated recoup index
- Dirty-read (DR) protect mode
- Key paths
- Properties
- Read-only attribute
- User class ID.

NAMe
displays the class name of the specified collection.

PARTs
displays the class names of the parts of the specified collection.

RECLaim
reclaims a collection marked for deletion.

RECONstruct
verifies and rebuilds the internal structures of a collection, and displays a list of all of the actions taken on the collection and any conditions that cannot be resolved.

Note: If multiple chains are corrupt, the ZBROW COLLECTION command with the RECONSTRUCT parameter specified may fail. For example, ZBROW COL RECONSTRUCT *colname* DIRECTORY and ZBROW COL RECONSTRUCT *colname* KEYS will both fail with a 0200DC system error if the allocated data chain is corrupt.

If you enter ZBROW COL RECONSTRUCT and it fails, enter a ZBROW COLLECTION command specifying the VALIDATE parameter for the collection and determine which chains are corrupt. Depending on the response to the ZBROW COLLECTION command with the VALIDATE parameter specified, you may either have to enter a ZBROW COLLECTION command with the RECONSTRUCT parameter specified for a different chain, or you may have to manually repair one of the chains using the ZAFIL command before attempting to enter the ZBROW COLLECTION command with the RECONSTRUCT parameter specified.

DATA
uses the internal mapping of a collection to rebuild the data chain.

KEYS
uses the data chain of a collection to rebuild the internal key (index) structure.

DIRECTory
uses the data chain and internal key (index) structure of a collection to rebuild the internal directory structure.

RESTore
restores a previously captured collection. Collections cannot be captured or restored using blocked tapes.

origname
is the name of the previously captured collection.

DDname-*ddname*
restores the data definition with the collections that will be displayed, where *ddname* is the name of the data definition.

SHADOW
specifies the use of shadowing for this restore. This parameter overrides the setting in the data definition.

VALidate
verifies the structure of the specified collection. If a key path name is not specified, the collection is verified without key path validation and the results of the verification are displayed. If the key path parameter is used, it verifies the structure of the specified key path and results in a dump if a structure that is not valid is found.

Additional Information

- Using this command to reconstruct a collection does not guarantee that the collection will be completely valid when this command has been completed. You may need to manually reconstruct the collection.
- Instead of using the EMPTY parameter with this command, you can use the ZOODB CHANGE command to set the data store to use immediate deletes.

Examples

The attributes of the BANK1_DS inventory collection are displayed in the following example.

Note: All fields displayed by this command are internal to TPFCS and are subject to change.

In this example, the ZBROW QUALIFY command is shown only to make the example more complete. You only have to enter the ZBROW QUALIFY command once. You do not have to enter the command again until the target data store needs to change to another data store.

```
User:   ZBROW QUAL SET DS-BANK1_DS

System: BROW0607I 08.36.31 QUALIFICATION PROCESSING COMPLETED

User:   ZBROW COL ATTR DS_INVENTORY

System: BROW0602I 10.11.38 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0406I 10.11.38 COLLECTION ATTRIBUTES DISPLAY
        ATTRIBUTE                           VALUE
        COLLECTION CLASS NAME - OIDinventory
          PART  NAME  **********            OIDentry
        CLASS NAME  **                      OBJECT
        OBJECT_ID                           00000034
        OBJECT_LGH                          00000742
        OBJECT_SEQ_CTR                      0000000F
        CLASS NAME  **                      ObjectPart
        OBJ_Part_CHANGE                     00
        OBJ_Part_RESERVE2                   000000
        OBJ_Part_PartID                     00000000
        OBJ_Part_OIE                        00812288
        CLASS NAME  **                      OIDentry
        OIDentry_OID                        0002FC16 AF44D6F9 E3D7C6C4 C2404040
        ****                                1802F078 1802F079 00000000 00000000
        OIDentry_Class_Value                00000000 00000000 00000000 00000000
        ****                                00000000 00000000 00000000 00000000
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

Collection MY_BROWSE is captured in the following example.

```
User:   ZBROW COL CAPT MY_BROWSE

System: COSK0079A 14.39.19 *CP* HPN    MOUNT ARB TAPE FOR OUTPUT

User:   ZTMNT ARB F32 AO BP

System: COTM0008W 14.40.33 TMNT HPN    DEVICE F32 VSN A00139
        UNEXPIRED FILE OVERWRITTEN
        COTM0310I 14.40.33 TMNT HPN    TAPE ARB MOUNTED ON DEVICE F32
                  VSN A00139 G0011 S0001 F38K     SL  NOBLK  NOCOMP
        COTC0300A 14.41.30 TCLS HPN    REMOVE ARB FROM DEVICE F32
                  VSN A00139 G0011 S0001 F38K     SL  NOBLK  NOCOMP
        BROW0650W 14.39.19 BROWSER NOT QUALIFIED, TPFDB ASSUMED
        BROW0602I 14.39.19 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0421I 14.41.30 COLLECTION MY_BROWSE SUCCESSFULLY CAPTURED
        BROW0419I 14.41.30 CAPTURE COMPLETED
```

Collection OLDCOL is copied in the following example to a new collection called NEWCOL.

```
User:   ZBROW COL COPY OLDCOL NEWCOL

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0303I 08.38.57 COLLECTION NAME SUCCESSFULLY DEFINED
        BROW0417I 08.38.57 COLLECTION COPIED SUCCESSFULLY
        BROW0411I 08.38.57 COPY COMPLETED
```

Collection OLDCOL is copied using the specified data definition to a new collection called NEWCOL in the following example.

```
User:   ZBROW COL COPYDD OLDCOL NEWCOL MYDD

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0303I 08.38.57 COLLECTION NAME SUCCESSFULLY DEFINED
        BROW0411I 08.38.57 COPY COMPLETED
```

Collection USER_110154687 is deleted in the following example.

```
User:   ZBROW COL DELETE USER_110154687

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0415I 08.38.57 COLLECTION MARKED FOR DELETION
        BROW0412I 08.38.57 DELETE COMPLETED
```

The following example causes TPFCS to dump the contents of collection DS_INVENTORY to a printer.

```
User:   ZBROW COL DUMP DS_INVENTORY

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0410I 08.38.57 BROWSE OF COLLECTION COMPLETED
```

All pending deletes are processed for data store BANK1_DS and all entries are removed from the DS_DELETED collection in the following example.

```
User:   ZBROW COL EMPTY DS_DELETED

System: BROW0602I 11.15.33 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0415I 11.15.33 COLLECTION HAS BEEN EMPTIED
        BROW0426I 11.15.33 EMPTY PROCESSING COMPLETED
```

In the following example, collection OLDCOL is migrated to a new collection called NEWCOL, which uses new pools in its current format.

```
User:   ZBROW COL MIGRATE OLDCOL NEWCOL

System: BROW0602I 11.37.52 BROWSER QUALIFIED FOR DSNAME OLDCOL
        BROW0303I 11.37.52 COLLECTION NAME SUCCESSFULLY DEFINED
        BROW0423I 11.37.52 COLLECTION MIGRATED SUCCESSFULLY
        BROW0424I 11.37.52 MIGRATION PROCESSING COMPLETED
```

The class name of the BANK1_DS inventory collection is displayed in the following example.

```
User:   ZBROW COL NAME DS_INVENTORY

System: BROW0602I 10.09.29 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0407I 10.09.30 COLLECTION NAME DISPLAY
        COLLECTION CLASS NAME - OIDinventory
        BROW0410I 10.09.30 BROWSE OF COLLECTION COMPLETED
```

The parts of the BANK1_DS inventory collection are displayed in the following example.

```
User:   ZBROW COL PARTS DS_INVENTORY

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0403I 08.38.57 COLLECTION PARTS DISPLAY
        PART    NAME
        0    OIDentry
        1    OIDinventory
        2    MemKey
        BROW0410I 08.38.57 BROWSE OF COLLECTION COMPLETED
```

The BANK1_DS inventory is being reclaimed in the following example.

```
User:   ZBROW COL RECLAIM DS_INVENTORY

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0413I 08.38.57 RECLAIM COMPLETED
```

In the following example, the structure of collection MYCOLLECT is verified and the internal directory structure of the collection is automatically reconstructed.

```
User:   ZBROW COL RECONSTRUCT MYCOLLECT DIRECTORY

System: BROW0650W 12.41.48 BROWSER NOT QUALIFIED, TPFDB ASSUMED
        BROW0602I 12.41.48 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0404I 12.41.49 RECONSTRUCTION REPORT DISPLAY
                           PID: 0002FC16 AF4469DD E3D7C6C4 C2404040  _
                                1802F104 1802F105 00000000 00000000

        ACTION ID  DESCRIPTION
        RECN1100 - ALLOCATED DIRECTORY CHAIN RECONSTRUCTED.
                   CHAIN LENGTH IS 00000001
        END OF DISPLAY
```

Collection MY_BROWSE is restored in the following example.

```
User:   ZBROW COL RESTORE MY_BROWSE HIS_BROWSE

System: TPXD0001A 14.42.13 LOAD TAPE VOLUME A00139 INTO DRIVE 0f32 MOUNTED AS ARA
        COSK0079A 14.42.13 *CP* HPN    MOUNT ARA TAPE FOR INPUT

User:   ZTMNT ARA F32 AI

System: COTM0310I 14.42.57 TMNT HPN    TAPE ARA MOUNTED ON DEVICE F32
        VSN A00139 G0012 S0001 F38K     SL  NOBLK
        BROW0650W 14.42.13 BROWSER NOT QUALIFIED, TPFDB ASSUMED
        BROW0602I 14.42.13 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0422I 14.43.57 COLLECTION MY_BROWSE SUCCESSFULLY RESTORED AS HIS_BROWSE
        BROW0420I 14.43.57 RESTORE COMPLETED
```

In the following example, the structure of collection DICT2 is verified.

```
User:      ZBROW COL VALIDATE DICT2

System:    BROW0602I 09.29.26 BROWSER QUALIFIED FOR DSNAME TO2KEYP
           BROW0402I 09.29.32 VALIDATION REPORT DISPLAY
                              PID: 0200FC16 B1ADD50F E3D6F2D2 C5E8D740
                                   1800E0B8 00000000 00000000 00000000

           ACTION ID  DESCRIPTION
           VALD0000 - VALIDATE FOUND NO STRUCTURAL ERRORS
                      ON ALLOCATED DIRECTORY CHAIN.
           VALD0000 - VALIDATE FOUND NO STRUCTURAL ERRORS
                      ON ALLOCATED DATA CHAIN.
           VALD0000 - VALIDATE FOUND NO STRUCTURAL ERRORS
                      ON ALLOCATED KEY CHAIN.
           VALD0000 - VALIDATE FOUND NO STRUCTURAL ERRORS
                      ON CONTROL CHAIN.
           END OF DISPLAY
```

Related Information

- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See “ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition” on page 1083 for more information about the ZOODB CHANGE command.

- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See *Messages (System Error and Offline)* and *Messages (Online)* for a list of the possible action codes that result when you specify the RECONSTRUCT or VALIDATE parameter.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW DISPLAY–Display Information about a Collection or Its Contents

Use this command to display information about a collection and to display the contents of a collection.

Requirements and Restrictions

- You must initialize TPF collection support (TPFCS) and define the target data store before you enter this command.
- The TPF system must have get file storage (GFS) pools enabled before you can use this command. Pools are not enabled until the system is cycled to CRAS state or above.
- You must enter the ZBROW QUALIFY command with the appropriate parameters specified to qualify the target element in a collection before you enter this command with the USEQUAL parameter specified. The appropriate qualification parameters will be used based on the collection type being accessed. See the collection support section in *TPF Database Reference* for more information about collection types.

Format

```
►►──ZBROW DISplay── ──────────────────────────────────────────────────────────►

►──┬─COLlection ──┬─colname─┬───────┬──────────────────┬─────────────┬───────►◄
   │              │         └─ ──PRT┘                  │             │
   │              └─ ──ALL── ──DSname-dsname─┬───────┬─┘             │
   │                                         └─ ──PRT┘               │
   ├─┬─ELEment─┬─ ──colname─┬─┬─────────────────────┬─┤ Options 1 ├─┬┤
   │ └─ENTry───┘            │ ├─ ──INDEX-index──────┤               ││
   │                        │ └─ ──VALue-─┬─decval──┤               ││
   │                        │             └─X%hexval┘               ││
   │                        └─ ──USEQual────────────────────────────┘│
   ├─FA── ──farf─────────────────────────────────────────────────────┤
   └─RRN── ──colname── ──rrn─────────────────────────────────────────┘
```

Options 1:

```
   ┌─ ──RSA-0.10──────────────────┐
├──┼──────────────────────────────┼──────────────────────────────────────────┤
   └─ ──RSA-displacement.length───┘
```

COLlection
: displays the names and values of all the attributes of the specified collection that are set by the user, such as data length, key length, collection type, and so on.

colname
: is the assigned name of the collection.

ALL
: displays all the collection names for the specified data store (DS).

PRT
sends results to the printer instead of displaying the values.

DSname-*dsname*
specifies the data store with the collections that will be displayed, where *dsname* is the name of the data store.

ELEment
displays the specified element of the collection. The data is displayed in hexadecimal format and in EBCDIC translation. If you do not specify additional parameters with this parameter, the first 16 bytes of the first element are displayed.

ENTry
has the same functionality as the ELEMENT parameter and is provided for compatibility. Use the ELEMENT parameter instead of the ENTRY parameter.

INDEX-*index*
displays the element of the collection corresponding to the index specified, where *index* is a 1-based long value. For array and sequence collections, 1 represents the first element; for logs and keyed logs, 1 represents the oldest element. For other collection types, the index will represent the element at the specified cursor position. The order established by the cursor is collection dependent and can vary.

VALue
displays the element of the collection whose key corresponds to the specified value. This is used for key bag, key set, key sorted bag, key sorted set, sorted bag, and sorted set collections.

decval
is a character string.

X%*hexval*
is a hexadecimal value.

USEQual
displays the element of the collection corresponding to the values saved from a previously entered ZBROW QUALIFY command. The data will be displayed starting at the position specified by the RSA parameter for a length indicated by the DLEN parameter. If DLEN was not specified, the data will be displayed for a length equal to the length of the DATA value. If DATA was not specified, the first 16 bytes will be displayed.

RSA-*displacement.length*
displays the relative starting address in the specified element for a specified length, where *displacement.length* is the hexadecimal displacement and length of the element to display. The display will show data rounded down and up to the nearest X'10' bytes. If the displacement plus the length is greater than the entry size, the display will end when the end of the entry is reached. The maximum displacement length is X'FFFF' bytes.

FA
displays TPF collection support (TPFCS) information about the record.

farf
is the 8- or 16-digit hexadecimal file address in file address reference format (FARF).

RRN
displays the directory entry for a specific relative record number (RRN) of a collection.

rrn is the hexadecimal RRN value whose directory entry will be displayed.

Additional Information

None.

Examples

In some of the following examples, the ZBROW QUALIFY command is shown only to make the examples more complete.

In the following example, information about collections that belong to the BANK1_DS data store are dumped.

```
User:    ZBROW QUAL SET DS-BANK1_DS

System:  BROW0607I 08.36.31 QUALIFICATION PROCESSING COMPLETED

User:    ZBROW DISP COL ALL DS-BANK1_DS PRT

System:  BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
         BROW0410I 08.38.57 BROWSE OF COLLECTION COMPLETED
```

The BANK1_DS inventory collection is displayed in the following example.

```
User:   ZBROW DISP COL DS_INVENTORY

System: BROW0602I 10.11.13 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0405I 17.56.40 COLLECTION DISPLAY
         PID - 0202FC16AF8601A4C6C1D74BC4E24040
               1802F4971802F4980000000000000000
            COLLECTION CLASS NAME - OIDinventory
            COLLECTION TYPE NAME - DICTIONARY
            COLLECTION CREATE TIME - AF8601A4B4CFA004
            COLLECTION RECOUP INDEX NAME -  *NONE*
            NUMBER ENTRIES - 00000126  MAXIMUM RECORD LENGTH - 00000038
            KEY LENGTH  - 00000010
            RESIDENCY TYPE - EXTENDED
            NUMBER OF RECORDS USED 0000000D
        END OF DISPLAY
        BROW0410I 17.56.41 BROWSE OF COLLECTION COMPLETED
```

An entry in the BANK1_DS inventory collection is displayed in the following example.

```
User:    ZBROW DISP ELEMENT DS_INVENTORY

System:  BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
         BROW0408I 08.38.57 COLLECTION ELEMENT DISPLAY
         CLASS ID 0000007C SEQ CTR 00000000 LGH 00000050
         00000000- F5F0F4F0 F0F1F1F0 5C5C5C5C 5C5CC1C2 50400110 ******AB
         END OF DISPLAY - ZEROED LINES NOT DISPLAYED
         BROW0410I 08.38.57 BROWSE OF COLLECTION COMPLETED
```

The contents of the DS_USER_DICT values are displayed in the following example.

```
User:   ZBROW DISP ELEMENT DS_USER_DICT VAL-X%D9C5C3D6E4D76DE3D7C6D2C5E86D RSA-0.50

System: BROW0602I 16.23.38 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW0408I 16.23.38 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000000 LGH 00000040
        00000000- 00000000 00000000 00000000 40404040 ........ ....
        00000010- 000001F4 000005AD 00000064 000000C8 ...4.... .......H
        00000020- 0002FC16 AF098124 E3D7C6C4 C2404040 ......a. TPFDB
        00000030- 1802DD83 1802DD84 00000000 00000000 ...c...d ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
        BROW0410I 16.23.38 BROWSE OF COLLECTION COMPLETED
```

The element inside collection ARRAYL1 that corresponds to the existing qualification is displayed in the following example.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 12.19.28 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 12.19.28 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA LENGTH  - 40
        RSA          - 0
        BROW0607I 12.19.28 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW DISPLAY ELEMENT ARRAYL1 USEQUAL

System: CSMP0097I 12.19.43 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 12.19.43 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 12.19.43 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000001 LGH 0000003C
         00000000- E2E8E2E3 C5D440E3 C5E2E340 C4C1E3C1 SYSTEM T EST DATA
         00000010- 40F0F0F0 F0F0F040 40404040 40C1D9D9  000000       ARR
         00000020- C1E8D3F1 40404040 40404040 40404040 AYL1
         00000030- C5D5C4D6 C6C4C1E3 C1000000          ENDOFDAT A...
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED+
        CSMP0097I 12.19.43 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0410I 12.19.43 BROWSE OF COLLECTION COMPLETED+

User:   ZBROW QUAL DISP ALL

System: CSMP0097I 12.22.07 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 12.22.07 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - NEWDATA
        RSA          - 1D
        BROW0607I 12.22.07 QUALIFICATION PROCESSING COMPLETED+

User:   ZBROW DISPLAY ELEMENT ARRAYL1 USEQUAL

System: CSMP0097I 12.21.41 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0602I 12.21.41 BROWSER QUALIFIED FOR DSNAME TO2SVTDS
        BROW0408I 12.21.41 COLLECTION ELEMENT DISPLAY
        CLASS ID 0000007D SEQ CTR 00000001 LGH 0000003C
         0000001D- C1D9D9C1 E8D3F1                     ARRAYL1
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED+
        CSMP0097I 12.21.41 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0410I 12.21.41 BROWSE OF COLLECTION COMPLETED+
```

In the previous examples, CLASS ID and SEQ CTR are internally used fields. LGH is the element size.

The following examples display information about the record whose file address is FA 18009D1C. All of the possible fields that can be shown in the file address display are listed here in alphabetic order, but are not shown in the display:

ASSOCIATED COLLECTION PID - *pid*
indicates the persistent identifier (PID) of the owning collection.

ASSOCIATED COLLECTION PREVIOUSLY DELETED
indicates the collection associated with this record was previously deleted by TPFCS.

COLLECTION DSNAME *dsname*
indicates the data store of the collection associated with this record, where *dsname* is the data store name.

CONTROL RECORD
indicates that the specified record is a control record. This message will also be displayed for overflow records for the first control record associated with the collection.

Note: For compact-resident collections, this message will be displayed even if the specified record contains a structure object.

DATA RECORD WITH *classname* OBJECT. RRN - *n*
indicates that this is a data record, where *classname* indicates the class name of the object used to package the collection data elements and *n* is the integer value, in decimal, of the relative record number (RRN).

DELPAGE RECORD
indicates that the specified record contains a DELPAGE object because TPFCS has deleted the record. The associated collection can still be active.

DIRECTORY ENTRY *directory_entry shadowed*
displays the contents of the directory entry that TPFCS uses to locate the specified record when processing its associated collection. The hexadecimal contents of the entry are displayed in place of *directory_entry*. If the directory entry indicates that the specified record is shadowed, the text SHADOWED will be displayed in place of the variable *shadowed* immediately after the directory entry contents. If the directory entry does not indicate that the specified record is shadowed, nothing is displayed immediately after the directory entry contents.

Directory entries are shown in standard or expanded format. Standard format indicates that file addresses are stored in 4-byte fields and expanded format indicates that file addresses are stored in 8-byte fields.

DIRECTORY RECORD WITH *classname* OBJECT
indicates that this is a directory record, where *classname* indicates the class name of the object used to package the directory entries.

ERROR READING OWNING CONTROL RECORD
indicates an input/output (I/O) error occurred when attempting to read the owning control record.

ERROR READING RECORD
indicates the file address that is supplied appears valid, but an I/O error occurred when attempting to read the record to obtain additional information.

FILE ADDRESS SPECIFIED NOT VALID
indicates the file address specified was not a valid TPF FARF file address.

FIXED FILE RECORD
indicates that the file address is a TPF fixed file record. TPFCS only uses pool file records to represent specific collections.

KEY RECORD WITH *classname* OBJECT. RRN - *n*
indicates that this is a key record, where *classname* indicates the class name of

the object used to package the key entries and *n* is the integer value, in decimal, of the relative record number RRN.

OWNING CONTROL RECORD FA - *farf1*
: indicates the file address of the owning control record, where *farf1* is an 8- or 16-digit hexadecimal FARF address.

OWNING CONTROL RECORD FA NOT VALID
: indicates the file address of the owning control record is not a valid TPF FARF address.

OWNING CONTROL RECORD NOT VALID
: indicates that the control record was read successfully, but is not valid for one of the following reasons:

- The control record is a fixed file record.
- The control record is a pool record, but not the size used by TPFCS.
- Even though the control record is a pool record of the correct size, it does not contain a control record object header at the correct displacement in the record.

OWNER ID FORMAT NOT VALID
: indicates the format of the owner identifier (ID) field in the record is not valid and, therefore, the file address of the owning control record contained in this field may be corrupt. As a result, any additional information about the owning control record or the collection associated with the file address you specified may be inaccurate.

POOL FILE RECORD
: indicates that the file address is a valid pool file record.

POOL FILE RECORD CANNOT BELONG TO TPFCS
: indicates the specified record cannot be used by TPFCS to represent a collection because it is not the correct pool record size used by TPFCS.

RELPAGE RECORD
: indicates that the specified record contains a RELPAGE object because TPFCS has scheduled the record to be deleted. The associated collection can still be active.

STRUCTURE RECORD WITH *classname* OBJECT
: indicates that this is the structure record of an extended-resident collection, where *classname* indicates the class name of the structure object.

UNKNOWN RECORD TYPE
: indicates that it is not possible to determine how the record is used by TPFCS because the record does not contain familiar object header information at the correct displacement.

The following example shows the directory entry in standard format:

```
User:    ZBROW DISP FA 18009D1C

System:  BROW0701I 07.20.54 FILE ADDRESS INFORMATION DISPLAY FOR FA 18009D1C
               POOL FILE RECORD
               DATA RECORD WITH NDXDATAPAGE OBJECT.  RRN -        3
               OWNING CONTROL RECORD FA -  18009D15
               ASSOCIATED COLLECTION PID - 0001FC16B0BAE322E3D7C6C4C2404040
                                           18009D1500000000000000000000000
               COLLECTION DSNAME TPFDB
               DIRECTORY ENTRY 0000000018009D1C0000000000000000
         END OF DISPLAY

         BROW0702I 07.20.54 BROWSE OF FILE ATTRIBUTES COMPLETED
```

The following example shows the directory entry in expanded format:

```
User:    ZBROW DISP FA 18009D1C

System:  BROW0701I 07.20.54 FILE ADDRESS INFORMATION DISPLAY FOR FA 18009D1C
               POOL FILE RECORD
               DATA RECORD WITH NDXDATAPAGE OBJECT.  RRN -        3
               OWNING CONTROL RECORD FA -  18009D15
               ASSOCIATED COLLECTION PID - 0001FC16B0BAE322E3D7C6C4C2404040
                                           18009D1500000000000000000000000
               COLLECTION DSNAME TPFDB
               DIRECTORY ENTRY 00008000000000000018009D1C000000000000000000000000
         END OF DISPLAY

         BROW0702I 07.20.54 BROWSE OF FILE ATTRIBUTES COMPLETED
```

The examples that follow show the directory entry for RRN E7D. The following example shows the directory entry in standard format:

```
User:     ZBROW DISP RRN DS_INVENTORY E7D

System:   BROW0602I 10.11.26 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0703I 10.11.26 DIRECTORY - 00000040 1800D2DF 1800D2E0 00000000
          BROW0410I 10.11.26 BROWSE OF COLLECTION COMPLETED
```

The following example shows the directory entry in expanded format:

```
User:     ZBROW DISP RRN DS_INVENTORY E7D

System:   BROW0602I 10.11.26 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0703I 10.11.26 DIRECTORY - 00008040 00000000 1800D2DF 00000000
                                         1800D2E0 00000000
          BROW0410I 10.11.26 BROWSE OF COLLECTION COMPLETED
```

Related Information

- See “ZBROW ALTER–Alter the Contents or Access Mode of a Specified Collection” on page 111 for more information about the ZBROW ALTER command.
- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW KEYPATH–Add, Display, or Remove a Key Path

Use this command to add, display, or remove an alternate key path that has been defined for a collection.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store (DS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.
- This function is only supported for persistent collections that support key paths. It is ***not*** supported for temporary collections.

Format

```
►►──ZBROW KEYPath── ──────────────────────────────────────────────────────►

►──┬─ADD── ──colname── ──keypathname── ──displacement── ──length─┬───────►◄
   ├─DISPlay── ──colname──┬─ ──ALL──────────┬─────────────────────┤
   │                      └─ ──keypathname──┘                     │
   └─REMove── ──colname── ──keypathname───────────────────────────┘
```

ADD
: creates a key path on the key path field that is specified by the displacement and length values for the specified collection.

colname
: is the 1- to 32-character alphanumeric name of the target collection.

keypathname
: is the 1- to 16-character alphanumeric name of the key path that will be added, displayed, or removed.

displacement
: specifies the 0-based hexadecimal displacement to the key path field in the data element.

length
: specifies the hexadecimal length of the key path field.

DISPlay
: displays information about the defined key paths for the specified collection.

ALL
: displays all the defined key paths for the specified collection.

REMove
: removes the specified key path for the collection.

Additional Information

- TO2_PRIME_KEYPATH is the reserved name of the primary key path and ***cannot*** be assigned to an alternate key path.

- The ZBROW KEYPATH command ***cannot*** be used to manage or display the primary key path.
- The key path resources of a collection that have been removed will be returned to the TPF system.

Examples

In the following examples, the ZBROW QUALIFY command with the SET parameter specified was issued to set the data store qualification to FAP.BSS before the examples were created.

A key path is added to the DICT1 key sorted set collection in the following example. The key path will be built using the values in the field starting at displacement 1 for a length of 8 characters for every data element in the collection.

```
User:      ZBROW KEYPATH ADD DICT1 INDEX1 1 8

System:    BROW0602I 12.38.16 BROWSER QUALIFIED FOR DSNAME FAP.BSS
           BROW0901I 12.38.17 KEYPATH SUCCESSFULLY ADDED
           BROW0910I 12.38.17 KEYPATH REQUEST COMPLETED
```

All the alternate key paths defined for the DICT1 key sorted set collection are displayed in the following example. The following fields are displayed:

KEYPATH NAME
: is the name of the key path.

DISPLACEMENT
: is the 0-based hexadecimal displacement of the key path field.

LENGTH
: is the hexadecimal length of the key path field.

COUNT
: is the decimal number of data elements accessible by using this key path.

status
: specifies whether or not the build is completed successfully for this key path.

```
User:      ZBROW KEYPATH DISP DICT1 ALL

System:    BROW0602I 12.46.09 BROWSER QUALIFIED FOR DSNAME FAP.BSS
           BROW0903I 12.46.09 KEYPATH INFORMATION DISPLAY
           KEYPATH NAME KEYPATH1          ID   1
           DISPLACEMENT 0001  LENGTH    8  COUNT         0
           BUILD COMPLETE
           KEYPATH NAME KEYPATH2          ID   2
           DISPLACEMENT 0020  LENGTH   50  COUNT         0
           BUILD COMPLETE
           END OF DISPLAY
           BROW0910I 12.46.09 KEYPATH REQUEST COMPLETED
```

The INDEX1 key path is removed from the DICT1 key sorted set collection in the following example.

```
User:     ZBROW KEYPATH REMOVE DICT1 INDEX1

System:   BROW0602I 12.39.25 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0902I 12.39.26 KEYPATH SUCCESSFULLY REMOVED
          BROW0910I 12.39.26 KEYPATH REQUEST COMPLETED
```

Related Information

- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW NAME–Alter or Display Collection Name Information

Use this command to define, display, or remove a name for a persistent identifier (PID) of a collection. Use the name in the other ZBROW messages in place of the PID. Use the ZBROW QUALIFY command to determine which data store (DS) to apply the request.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZBROW NAMe── ──┬─DEFine── ──colname── ──PID-pid─┬──────────────────────────►◄
                   ├─DISPlay── ──┬─colname─┬──────────┤
                   │             └─ALL─────┘          │
                   └─REMove── ──colname───────────────┘
```

DEFine
defines a name for a collection and associates it with a specified PID.

colname
is the 1- to 32-character name of the collection that you want to define, display, or remove.

PID-*pid*
specifies the PID, where *pid* is the 64-character hexadecimal PID value.

Note: You are not required to enter the trailing zeros for *pid*.

DISPlay
displays the PID associated with a collection name or, if specified with the ALL parameter, displays all defined collection names.

ALL
displays all defined collection names in the browser dictionary along with their associated PID values.

REMove
removes the name of the collection definition from the TPFCS browser dictionary.

Additional Information

Whenever a data store is defined using the ZOODB DEFINE command with the DS parameter specified, the following browser names (see the example for ZBROW QUAL SET DS-BANK1_DS) are also added to the browser dictionary of the data store:

DS_BROWSE
Name of the browser dictionary of the data store.

DS_DELETED
Name of the deleted PID collection of the data store.

DS_INVENTORY
Name of the inventory collection of the data store. The inventory collection name is not defined if NOINVENTORY was specified on the latest `TO2_createDS` or `TO2_changeDS` request.

DS_RECOUP
Name of the recoup index collection of the data store.

DS_SYSTEM_DICT
Name of the system dictionary of the data store.

DS_USER_DICT
Name of the user dictionary of the data store.

Examples

The USER_ACCOUNTS name is assigned to the data store user account collection of the BANK1_DS collection in the following example.

In this example, the ZBROW QUALIFY command is shown only to make the example more complete. You only have to enter the ZBROW QUALIFY command once. You do not have to enter the command again until the target data store needs to change to another data store.

```
User:   ZBROW QUAL SET DS-BANK1_DS

System: BROW0607I 08.36.31 QUALIFICATION PROCESSING COMPLETED

User:   ZBROW NAME DEF USER_ACCOUNTS PID-0200FC19AD58DB59C2C1D5D2F16DC4E2980081DD

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0303I 08.38.57 COLLECTION NAME SUCCESSFULLY DEFINED
        BROW0310I 08.38.57 NAME DEFINITION REQUEST COMPLETED
```

The PID of the BANK1_DS inventory collection is displayed in the following example.

```
User:   ZBROW NAME DISP DS_INVENTORY

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0301I 08.38.57 NAME DEFINITION DISPLAY

        NAME - DS_INVENTORY
        PID  - 0200FC19 AD58DB59 C2C1D5D2 F16DC4E2
               980081DD 00000000 00000000 00000000
        END OF DISPLAY

        BROW0310I 08.38.57 NAME DEFINITION REQUEST COMPLETED
```

All the collection names and corresponding PIDs in the browser dictionary for data store BANK1_DS are displayed in the following example.

```
User:   ZBROW NAME DISP ALL

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0301I 08.38.57 NAME DEFINITION DISPLAY

        NAME - USER_ACCOUNTS
        PID  - 0200FC19 AD58DB59 C2C1D5D2 F16DC4E2
               980081DD 00000000 00000000 00000000
        NAME - DS_BROWSE
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA9E 00000000 00000000 00000000
        NAME - DS_DELETED
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA9C 00000000 00000000 00000000
        NAME - DS_INVENTORY
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA99 00000000 00000000 00000000
        NAME - DS_RECOUP
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA9D 00000000 00000000 00000000
        NAME - DS_SYSTEM_DICT
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA9B 00000000 00000000 00000000
        NAME - DS_USER_DICT
        PID  - 0200FC19 AEC8FD8E C2C1D5D2 F16DC4E2
               180BDA9A 00000000 00000000 00000000
        END OF DISPLAY

        BROW0310I 08.38.57 NAME DEFINITION REQUEST COMPLETED
```

The USER_ACCOUNTS collection name is removed in the following example.

```
User:   ZBROW NAME REM USER_ACCOUNTS

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0304I 08.38.57 COLLECTION NAME SUCCESSFULLY REMOVED
        BROW0310I 08.38.57 NAME DEFINITION REQUEST COMPLETED
```

The PID of the MATIP_DS database is displayed in the following example.

```
User:   ZBROW NAME DISP HOSTNAMES

System: BROW0602I 07.33.37 BROWSER QUALIFIED FOR DSNAME MATIP_DS
        BROW0301I 07.33.37 NAME DEFINITION DISPLAY

        NAME - HOSTNAMES
        PID  - 0200FC19 B2C4CB4B D4C1E3C9 D76DC4E2
               18003B5C 00000000 00000000 00000000
        END OF DISPLAY

        BROW0310I 07.33.37 NAME DEFINITION REQUEST COMPLETED
```

Related Information

- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*

– *TPF Database Reference*.

ZBROW PATH–Display Path Information for a Collection Structure

Use this command to display path information for a specific structure of a collection. Use the ZBROW PATH command to display the following:

- The actual path information for keys and relative record numbers (RRNs)
- The actual starting location of an array element or a relative byte address (RBA) on file.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store (DS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.
- This function is supported only for persistent extended collections. The ZBROW PATH command does not support persistent compact or temporary collections.

Format

```
►►──ZBROW PATH── ─────────────────────────────────────────────────────────►

►──┬─KEY── ──colname─┬──────────────────────────┬─ ──KEY-─┬─deckey──┬─┬──────►◄
   │                 └─ ──KEYPath-keypathname─┘         └─X%hexkey─┘ │
   ├─INDEX── ──colname── ──index─────────────────────────────────────┤
   ├─RBA── ──colname── ──rba─────────────────────────────────────────┤
   └─RRN── ──colname── ──rrn─────────────────────────────────────────┘
```

KEY
: displays path information for the index structure for the specified key path and key.

colname
: is the 1- to 32-character alphanumeric name of the target collection.

KEYPath-*keypathname*
: specifies the key path, where *keypathname* is a 1- to 16-character alphanumeric name of the key path that will be traversed for the path information.

KEY
: specifies the key, where:

  key
  : is a 1- to 128-byte string of the key whose path information will be displayed.

  X%*hexkey*
  : is a 1- to 128-byte hexadecimal string.

  You can specify the beginning characters of a key if a partial search is desired.

deckey
: is a character string.

INDEX
displays location information of the entry for the specified index value.

index
is the decimal index value of the entry whose location will be displayed.

RBA
displays byte location information for the specified RBA value.

rba
is the hexadecimal RBA value whose location will be displayed.

RRN
displays path information for the RRN structure for the specified RRN value.

rrn specifies the hexadecimal RRN value whose path information will be displayed.

Additional Information

For nonunique collections and partially specified keys, the path information for the first key found will be retrieved.

Examples

In the examples that follow, the ZBROW QUALIFY command with the SET parameter specified was issued to set the data store qualification to FAP.BSS before the examples were created.

The following example shows the path display for a DS_INVENTORY partial key of 18003465. The following fields may be included in displays of key path entries:

ROOT
is the display of the root index record. It shows the key length, RRN, and highest key value of the root index record.

KEYLENGTH
specifies the decimal key length of the keys in the index for the key path.

Note: The key length shown in the display is the internal length of the key, not the user-defined length of the key.

RRN
is the hexadecimal value of the RRN of the root.

TYPE
specifies the type of entry being displayed, where:

I is the display of a high key entry that points to another index record in the index path.

D is the display of a high key entry that points to the data record at the end of the key path.

NEXT RRN
is the hexadecimal value of the next RRN in the key path.

The first pair of lines in the display show the entry for the root index record of the index structure. These lines show the first 16 bytes of the highest key of the collection itself. The next pair of lines show the actual high key of the path that will be followed through the index structure for the specified key. This is the entry from the root, and the specified RRN is the RRN of the next index record in the path. The next entry is the entry from that index record and contains the high key of the data record that would contain the specified data entry for the given key if it exists.

Because there is only one TYPE-I display line, the index structure is only two records deep (the root record and the intermediate index record), and the data record would be the third record read.

Because keys can be a maximum of 256 bytes in length and, therefore, would not fit on an output line, the display shows only the first 16 bytes of both the highest key in the root and the first key in the path. Thereafter, the display will show the first 14 bytes of the following keys starting at the first character that is different from the previous key. The first number in the display line is the displacement of the first different character. So, for the TYPE-D line, the displacement of the first different character is 2 and then the next 14 characters are displayed. If no characters are different, the displacement value will be 0 and the full first 16 characters will be displayed.

```
User:   ZBROW PATH KEY DS_INVENTORY KEY-X%18003465
System: BROW0602I 16.45.53 BROWSER QUALIFIED FOR DSNAME FAP.BSS
        BROW0801I 16.45.54 PATH FOR SPECIFIED KEY FOR COLLECTION DS_INVENTORY

        ROOT  KEYLENGTH - 16  RRN-0
         00   1801F26F 00000000 00000000 B031C597     ..2?..........Ep
        TYPE - I  NEXT RRN-24C
         00   18003526 00000000 00000000 B031477B     ...............#
        TYPE - D  NEXT RRN-273
         02       3470 00000000 00000000 B03146E7       .............X
        END OF DISPLAY

        BROW0410I 16.45.54 BROWSE OF COLLECTION COMPLETED
```

The following example shows the path display for a DS_INVENTORY partial key of the highest key value of 1801F26F:

```
User:   ZBROW PATH KEY DS_INVENTORY KEY-X%1801F26F
System: BROW0602I 16.58.40 BROWSER QUALIFIED FOR DSNAME FAP.BSS
        BROW0801I 16.58.40 PATH FOR SPECIFIED KEY FOR COLLECTION DS_INVENTORY

        ROOT  KEYLENGTH - 16  RRN-0
         00   1801F26F 00000000 00000000 B031C597     ..2?..........Ep
        TYPE - I  NEXT RRN-261
         00   1801F26F 00000000 00000000 B031C597     ..2?..........Ep
        TYPE - D  NEXT RRN-2F1
         00   1801F26F 00000000 00000000 B031C597     ..2?..........Ep
        END OF DISPLAY

        BROW0410I 16.58.40 BROWSE OF COLLECTION COMPLETED
```

The index structure for the DS_INVENTORY collection for the key starting with hexadecimal string B0309868 using the TIME_INDEX3 key path is displayed in the following example.

```
User:     ZBROW PATH KEY DS_INVENTORY KEYP-TIME_INDEX3 KEY-X%B0309868
System:   BROW0602I 10.06.07 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0801I 10.06.07 PATH FOR SPECIFIED KEY FOR COLLECTION DS_INVENTORY
          TYPE - I  KEYLENGTH - 12  RRN-E7D
           00   B030ACE8 B1481B56 03128A01     ...Y........
          TYPE - D  KEYLENGTH - 12  RRN-DD8
           02       9A85 B1481B14 DA8C6C01       .e......%.
          END OF DISPLAY

          BROW0410I 10.06.07 BROWSE OF COLLECTION COMPLETED
```

The RRN of the data record and the displacement in the data object of the first byte of the array entry for index 120 is displayed in the following example.

```
User:     ZBROW PATH INDEX ARRAY1 120

System:   BROW0602I 10.15.14 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0803I 10.15.14 INFORMATION FOR INDEX 120 FOR COLLECTION ARRAY1

          RRN - 6  DISPLACEMENT - 5C4
          END OF DISPLAY

          BROW0410I 10.15.14 BROWSE OF COLLECTION COMPLETED
```

The RRN of the data record and the displacement in the data object for RBA 12AB0 is displayed in the following example.

```
User:     ZBROW PATH RBA BLOB1 12AB0

System:   BROW0602I 10.17.05 BROWSER QUALIFIED FOR DSNAME FAP.BSS
          BROW0804I 10.17.05 INFORMATION FOR RBA 12AB0 FOR COLLECTION BLOB1

          RRN - 12  DISPLACEMENT - EC4
          END OF DISPLAY

          BROW0410I 10.17.05 BROWSE OF COLLECTION COMPLETED
```

The RRN path for the DS_INVENTORY collection for RRN 8D is displayed in the following examples. The TYPE value describes the type of index entry, as follows:

TYPE-S
: is the directory entry of the structure record.

TYPE-B
: is a directory entry embedded within the structure object. A small number of directory entries are embedded in the structure and rest are kept in a directory tree.

TYPE-R
: is the directory entry of the root directory record.

TYPE-I
: is a directory entry that points to another directory record in the path.

TYPE-D
: is a directory entry that points to the directory record that contains the directory entry of the specified RRN.

Directory entries are shown in standard or expanded format. Standard format indicates that file addresses are stored in 4-byte fields and expanded format indicates that file addresses are stored in 8-byte fields.

The directory entries in the following example are shown in standard format.

```
User:   ZBROW PATH RRN DS_INVENTORY 8D

System: BROW0602I 17.12.33 BROWSER QUALIFIED FOR DSNAME FAP.BSS
        BROW0802I 17.12.33 PATH FOR RRN 8D FOR COLLECTION DS_INVENTORY

        TYPE - S             DISPLACEMENT - 4C
        DIRECTORY - 00000040 180000B5 180000B6 00000000
        TYPE - R             DISPLACEMENT - 1F4
        DIRECTORY - 00020040 1800069A 1800069B 00000000
        TYPE - I             DISPLACEMENT - 10
        DIRECTORY - 00000040 1801CE18 1801CE19 00000000
        TYPE - D             DISPLACEMENT - 7C0
        DIRECTORY - 00000040 1801C43A 1801C43B 00000000
        END OF DISPLAY

        BROW0410I 17.12.33 BROWSE OF COLLECTION COMPLETED
```

The directory entries in the following example are shown in expanded format.

```
User:   ZBROW PATH RRN DS_INVENTORY 8D

System: BROW0602I 17.12.33 BROWSER QUALIFIED FOR DSNAME FAP.BSS
        BROW0802I 17.12.33 PATH FOR RRN 8D FOR COLLECTION DS_INVENTORY

        TYPE - S             DISPLACEMENT - 4C
        DIRECTORY - 00008040 00000000 180000B5 00000000
                    180000B6 00000000
        TYPE - R             DISPLACEMENT - 1F4
        DIRECTORY - 00028040 00000000 1800069A 00000000
                    1800069B 00000000
        TYPE - I             DISPLACEMENT - 10
        DIRECTORY - 00008040 00000000 1801CE18 00000000
                    1801CE19 00000000
        TYPE - D             DISPLACEMENT - 7C0
        DIRECTORY - 00008040 00000000 1801C43A 00000000
                    1801C43B 00000000
        END OF DISPLAY

        BROW0410I 17.12.33 BROWSE OF COLLECTION COMPLETED
```

Related Information

- See “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about the ZBROW QUALIFY command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW PROPERTY–Alter or Display a Property

Use this command to define, delete, or display a property for the specified collection. When defining these properties (also referred to as attributes), you can:

- Define a new property and assign its value and mode
- Modify the value and mode of an existing property
- Delete a property
- Display the value and mode of a property.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store (DS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZBROW PROPerty── ─────────────────────────────────────────────────────────►

►──┬─DEFine── ──colname── ──propname── ──┬─CHAR───┬── ──VALue-value──┤ Options ├─┬────►◄
   │                                     ├─DOUBLE─┤                              │
   │                                     ├─LONG───┤                              │
   │                                     ├─STRING─┤                              │
   │                                     └─STRUCT─┘                              │
   ├─DELete── ──colname── ──┬─propname─┬──────────────────────────────────────────┤
   │                        └─ALL──────┘                                          │
   └─DISPlay── ──colname── ──┬─propname─┬─────────────────────────────────────────┘
                             └─ALL──────┘
```

Options:

```
   ┌─ ──MODE-NORMAL────────────┐
├──┼───────────────────────────┼──────────────────────────────────────────────┤
   └─ ──MODE-──┬─NOCHANGE─────┘
               ├─NODELETE─┤
               └─READONLY─┘
```

DEFine
: defines or modifies a property for the specified collection.

colname
: is the 1- to 32-character alphanumeric name of the collection you want to define, delete, or display.

propname
: is the 1- to 64-character alphanumeric name of the property you want to define, delete, or display.

CHAR
: specifies a 1-byte character. This character cannot be a space, dash (-), or equal sign (=).

DOUBLE
specifies an 8-byte nonnegative decimal value. The maximum value for a DOUBLE type is $2^{63}-1$. A decimal value is entered, but it is displayed as the internal representation of the number. This consists of a sign bit, a 7-bit hexadecimal characteristic, and a 24-bit hexadecimal fraction (significand), which is preceded by an implied decimal point. The number is calculated as:

```
(−1) × (sign bit) × (significand) × (Base^Exponent)
```

where: Exponent = characteristic − Bias, and the Bias=64 and the Base=16 in an S/370 environment.

LONG
specifies a 4-byte nonnegative decimal value. The maximum for a LONG type is $2^{31}-1$.

STRING
specifies a C character string that is up to 256 bytes. The string cannot contain any spaces, dashes, or equal (=) signs.

STRUCT
specifies a structure that is up to 1000 bytes.

VALue-*value*
specifies the value of the property. The value must be consistent with the type specified.

MODE
specifies one of the following modes for a property:

NORMAL
specifies that you can read, change, or delete the property mode with no restrictions.

NOCHANGE
specifies that the property can only be read and deleted. It cannot be changed.

NODELETE
specifies that the property mode can be read or changed but it cannot be deleted.

READONLY
specifies that the property mode can only be read. The property mode cannot be changed or deleted. A READONLY property is only deleted when the target collection is deleted.

DELete
deletes a property for the specified collection.

ALL
specifies one of the following:

- When you specify the DELETE parameter, the ALL parameter deletes all defined property names for a specific collection except for properties whose mode is NODELETE or READONLY.
- When you specify the DISPLAY parameter, the ALL parameter displays all defined property names for a specific collection.

DISPlay
displays either a definition of a property (including the type, mode, and value) or all defined property names for a specific collection. Numeric values are displayed in hexadecimal.

Additional Information

None.

Examples

The DICTIONARYNAME property is defined for the DICT1 collection and given a value of MYDICTIONARY in the following example.

In this example, the ZBROW QUALIFY command is shown only to make the example more complete. You only have to enter the ZBROW QUALIFY command once. You do not have to enter the command again until the target data store needs to change to another data store.

```
User:   ZBROW QUAL SET DS-BANK1_DS

System: BROW0607I 08.36.31 QUALIFICATION PROCESSING COMPLETED

User:   ZBROW PROPERTY DEFINE DICT1 DICTIONARYNAME STRING VAL-MYDICTIONARY

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0510I 18.56.22 PROPERTY DEFINITION REQUEST COMPLETED
```

The property type, mode, and value of the DICTIONARYNAME property for the DICT1 collection are displayed in the following example along with the PID of the property collection.

```
User:   ZBROW PROPERTY DISPLAY DICT1 DICTIONARYNAME

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0501I 18.56.50 PROPERTY DISPLAY
        PID  - 0200FC19 AEA2673D C6C1D74B C4C24040
               1801C43F 00000000 00000000 00000000
        NAME:
        DICTIONARYNAME
        TYPE - STRING   MODE - NORMAL
        VALUE - MYDICTIONARY
        BROW0510I 18.56.50 PROPERTY DEFINITION REQUEST COMPLETED
```

A list of all properties defined for the DICT1 collection and the PID of the property collection is displayed in the following example.

```
User:   ZBROW PROPERTY DISPLAY DICT1 ALL

System: BROW0602I 08.38.57 BROWSER QUALIFIED FOR DSNAME BANK1_DS
        BROW0501I 18.59.41 PROPERTY DISPLAY
        PID  - 0200FC19 AEA2673D C6C1D74B C4C24040
               1801C43F 00000000 00000000 00000000
        DICTIONARYNAME
        MYPROP
        BROW0510I 18.59.41 PROPERTY DEFINITION REQUEST COMPLETED
```

Related Information

- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*

– *TPF Database Reference*.

ZBROW QUALIFY–Qualify ZBROW for a Data Store

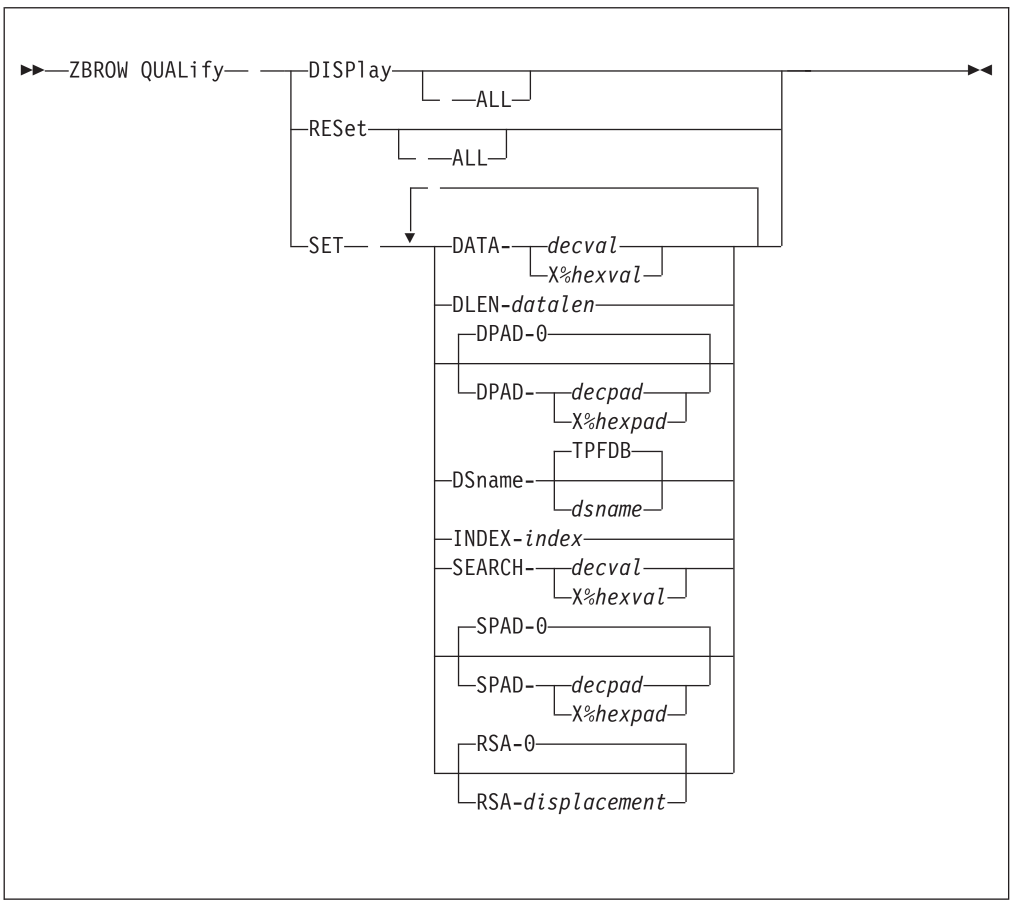
Use this command to specify to which data store and parameters subsequent ZBROW commands apply. With this command you can:

- Set the data store to be used on subsequent ZBROW commands
- Set the parameters for subsequent ZBROW ALTER commands
- Reset the parameters of the ZBROW qualification
- Display the current ZBROW qualification.

Requirements and Restrictions

- You must initialize TPF collection support (TPFCS).
- The TPF system must have get file storage (GFS) pools enabled before you can use this command. Pools are not enabled until the system is cycled to CRAS state or above.
- All qualification parameters entered are not verified until the qualification is used by a subsequent ZBROW command.

Format



DISPlay
displays the current ZBROW command qualification.

ALL
displays the current entire ZBROW command qualification when used with the

DISPLAY parameter. When used with the RESET parameter, all entries of the qualification are zeroed out, including the data store name.

RESet
zeros out all entries of the qualification except the data store name.

SET
sets or updates the qualification of the specified parameters.

DATA
specifies the data that will be used to retrieve from, or store into, a collection. The maximum input size is 108 characters.

decval
is a character string.

X%*hexval*
is a hexadecimal character string.

DLEN-*datalen*
specifies the hexadecimal length of the data being added, modified, or removed, where *datalen* is the length. The maximum data length is X'80' bytes. If no data length is specified, the length of the data entered is determined dynamically. If a length is specified that is less than the actual length of the data, an error will be generated on a subsequent command that attempts to use that data. If a length is specified that is greater than the actual length of the data, the data is padded at the end with the character specified by the DPAD parameter. If a delete operation is being performed on a binary large object (BLOB), this parameter is required and specifies the number of bytes to delete.

DPAD
specifies the character to use in data padding operations.

decpad
specifies the character to use in padding operations, where *decpad* is a single character.

X%*hexpad*
specifies the hexadecimal value to use in padding operations, where *hexpad* is a 1-byte value.

DSname-*dsname*
specifies the data store name that the ZBROW command will use for all subsequent ZBROW commands, where *dsname* is a 1- to 8-character data store name.

INDEX-*index*
specifies the index of the element to be added, modified, or deleted, where *index* is a 1-based decimal value. If you specify INDEX, you cannot specify SEARCH or SPAD. You can specify SEARCH and SPAD concurrently.

SEARCH
specifies the search field of the element to be added, modified, or deleted. If you specify INDEX, you cannot specify SEARCH or SPAD. You can specify SEARCH and SPAD concurrently. If the length of the search field specified is less than the collection search field size, SEARCH is padded at the end with the character specified by SPAD. The maximum size of the search field is 108 characters.

SPAD
specifies the character to use in SEARCH padding operations. If you specify INDEX, you cannot specify SEARCH or SPAD. You can specify SEARCH and SPAD concurrently.

RSA-*displacement*
specifies the starting address of data in an element or a BLOB as the target to be modified, where *displacement* is a 0-based starting hexadecimal displacement.

Additional Information

- ZBROW qualifications are associated with a specific terminal address (LNIATA) in the TPF complex. Therefore, each terminal address can specify its own unique qualification.
- The value for a particular qualification parameter stays in effect until the same terminal address enters either another value for that parameter or the RESET parameter. The default value for a parameter will not overwrite the existing value.

Examples

The ZBROW qualification is displayed in the following example.

```
User:   ZBROW QUAL DISP

System: BROW0602I 08.37.20 BROWSER QUALIFIED FOR DSNAME BANK1_DS
```

The entire ZBROW qualification is displayed in the following example.

```
User:   ZBROW QUAL DISP ALL

System: CSMP0097I 09.45.15 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 09.45.15 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        DATA         - TESTING
        DATA LENGTH  - 20
        DATA PAD     - G
        BROW0607I 09.45.15 QUALIFICATION PROCESSING COMPLETED+
```

The ZBROW qualification, except for the data store name, is reset in the following example.

```
User:   ZBROW QUAL RESET

System: CSMP0097I 09.05.50 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0607I 09.05.50 QUALIFICATION PROCESSING COMPLETED

User:   ZBROW QUAL DISP ALL

System: CSMP0097I 09.15.50 CPU-B SS-BSS  SSU-HPN  IS-01
        BROW0606I 09.15.50 BROWSER QUALIFICATION DISPLAY
        TERM ADDRESS - 10000
        DSNAME       - TO2SVTDS
        BROW0607I 09.15.50 QUALIFICATION PROCESSING COMPLETED
```

The ZBROW qualification is set to data store BANK1_DS in the following example.

```
User:   ZBROW QUAL SET DS-BANK1_DS

System: BROW0607I 08.36.31 QUALIFICATION PROCESSING COMPLETED
```

The ZBROW qualification is set to data store TO2SVTDS and the data is specified to be NEWDATA in the following example. This qualification can be used to allow the contents of a collection to be changed on a subsequent ZBROW ALTER command.

```
User:   ZBROW QUAL SET DS-TO2SVTDS DATA-NEWDATA

System: BROW0607I 08.53.59 QUALIFICATION PROCESSING COMPLETED
```

See "ZBROW ALTER–Alter the Contents or Access Mode of a Specified Collection" on page 111 for more examples.

Related Information

- See "ZOODB INIT–Initialize TPF Collection Support" on page 1097 for more information about initializing TPF collection support.
- See "ZOODB DEFINE–Define Data Store or Data Definition" on page 1086 for more information about defining a target data store.
- See "ZBROW ALTER–Alter the Contents or Access Mode of a Specified Collection" on page 111 for more information about the ZBROW ALTER command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBROW RECOUP–Manage Recoup Indexes

Use this command to manage recoup indexes, which describe the location of embedded file addresses and persistent identifiers (PIDs) in collections. With this command, you can:

- Create and delete recoup indexes
- Add and delete entries in a recoup index
- Display a recoup index
- Create and remove recoup index associations with collections.

Requirements and Restrictions

- You must initialize TPF collection support (TPFCS) and define the target data store (DS) before you enter this command.
- The TPF system must have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZBROW RECoup── ───────────────────────────────────────────────────────►

►──┬─ADD── ──indexname── ──ETOKen-─┬─charstr─┬─┤ Options 1 ├─────┬───────►◄
   │                               └─X%hexstr┘                  │
   ├─DEFine── ──indexname── ──┬─BLOB──────────┬──┬──────────────┬┤
   │                          ├─HETerogeneous─┤  └──DESC-text───┘│
   │                          └─HOMogeneous───┘                  │
   ├─DELete── ──indexname────────────────────────────────────────┤
   ├─DISPlay── ──┬─indexname─┬───────────────────────────────────┤
   │             └─ALL───────┘                                   │
   ├─LINK── ──colname── ──indexname──────────────────────────────┤
   ├─REMove── ──indexname── ──ETOKen-─┬─charstr──┬───────────────┤
   │                                  └─X%hexstr─┘               │
   └─UNLink── ──colname──────────────────────────────────────────┘
```

Options 1:

```
├──DISP-displ── ──ETYPe-─┬─FA──┬──┬─────────────────────────┬──────────┤
                         └─PID─┘  ├─ ──INDEX-decval─────────┤
                                  └─ ──KEY-─┬─charstr──┬────┘
                                            └─X%hexstr─┘
```

ADD
: adds an entry to an existing recoup index.

indexname
: is the 8-byte character name of the recoup index being managed. If you specify a character name that is less than 8 bytes, the name will be padded on the right with spaces.

ETOKen
specifies a unique 8-byte identifier for the given entry in the specified recoup index. If you specify a character name that is less than 8 bytes, the name will be padded on the right with spaces. If you specify a hexadecimal string that is less than 8 bytes, the string will be padded on the right with zeros.

charstr
is a character string.

X%*hexstr*
is a hexadecimal string.

DISP-*displ*
specifies the displacement into the user collection element for the user-embedded PID or file address, where *displ* is the 0-based hexadecimal displacement.

ETYPe
specifies one of the following embedded recoup reference type indicators:

FA
indicates that the index entry being added describes a 16-byte area that contains a file address.

See *TPF Database Reference* for more information about the format of the file addresses embedded in collections.

PID
indicates that the index entry being added describes a PID.

INDEX-*decval*
specifies the index to access the user collection element that contains the embedded recoup reference, where *decval* is the 1-based decimal value of the index. Either this parameter or the KEY parameter is needed for a heterogeneous collection.

KEY
indicates the value of a character key to access the user collection element that contains the embedded recoup reference. Either this parameter or the INDEX-*decval* parameter is needed for a heterogeneous collection.

DEFine
creates a recoup index.

BLOB
indicates that the user collection is a binary large object (BLOB) and therefore not homogeneous or heterogeneous.

HOMogeneous
indicates that all elements in the collection have the same displacement to the PID or file address and can be recouped the same way.

HETerogeneous
indicates that not all elements contain an embedded file address or PID, or the displacements are different (that is, not all elements can be recouped the same way).

DESC-*text*
describes the recoup index, where *text* is up to 256 alphanumeric characters.

DELete
deletes a recoup index.

DISPlay
displays a specified recoup index or a list of all defined recoup indexes from the qualified data store.

ALL
displays a list of all defined recoup indexes from the qualified data store.

LINK
associates a recoup index with a specified existing collection. Each time a collection is linked, it will overwrite previous settings without error.

colname
is the 1- to 32-character alphanumeric name of the collection whose recoup index association is being changed.

REMove
deletes an entry from a recoup index.

UNLink
removes the association between the specified collection and its recoup index. If the collection is not associated with a recoup index, no error will be indicated.

Additional Information

None.

Examples

An entry is added to recoup index INDEX001 in data store TESTX.DS in the following example. The entry has a key of TOKEN001 and indicates that all user collections associated with this recoup index are either homogeneous or a BLOB. If the collection is homogeneous, there is a PID at displacement 0 of each element; if the collection is a BLOB, there is a PID at displacement 0.

In this example, the ZBROW QUALIFY command is shown only to make the example more complete. You only have to enter the ZBROW QUALIFY command once; you do not have to enter the command again until the target data store needs to change to another data store.

```
User:   ZBROW QUAL SET DS-TESTX.DS

System: BROW0607I 10.56.31 QUALIFICATION PROCESSING COMPLETED

User:   ZBROW RECOUP ADD INDEX001 ETOK-TOKEN001 DISP-0 ETYPE-PID

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1011I 10.56.43 RECOUP INDEX ENTRY SUCCESSFULLY ADDED
```

Recoup index INDEX001 is created in the following example to describe a homogeneous collection.

```
User:   ZBROW RECOUP DEFINE INDEX001 HOMOG DESC-PRIMARY_INDEX

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1012I 10.56.43 RECOUP INDEX SUCCESSFULLY DEFINED
```

Recoup index INDEX001 is deleted in the following example.

```
User:   ZBROW RECOUP DELETE INDEX001

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1013I 10.56.43 RECOUP INDEX SUCCESSFULLY DELETED
```

The contents of recoup index INDEX001 in data store TPFDB are displayed in the example that follows. The display shows the recoup index header and each entry in the recoup index, where:

NAME
: is the name of the recoup index.

PID
: is the persistent identifier (PID) of the recoup index.

DESC
: is the description of a recoup index that was entered when the index was defined.

TYPE
: is one of the following:

  HOMOGENEOUS
  : indicates that all the elements of the collection that the recoup index is associated with have the same format.

  HETEROGENEOUS
  : indicates that the elements of the collection with which the recoup index is associated have different formats.

  IGNORED (BLOB)
  : indicates that the collection is a BLOB and the recoup index is not homogeneous or heterogeneous.

CONTROL
: indicates which processing restrictions apply, where NONE indicates that no specific processing restrictions apply.

TOKEN
: is the unique identifier for each entry in the recoup index, displayed in hexadecimal characters.

TYPE
: is one of the following:

  PID
  : indicates that the entry refers to an embedded PID in the collection elements.

  FA
  : indicates that the entry refers to an embedded file address in the collection elements.

DISPL
: is the displacement into the collection element for the embedded information. This is shown as a hexadecimal value.

The following fields, which are not shown in this example, are included with displays for heterogeneous recoup indexes:

ACCESS
: is one of the following:

INDEX
: indicates that the collection elements are accessed by a specified index.

KEY
: indicates that the collection elements are accessed by a specified key.

IDX
: is the index used to select the collection element for heterogeneous recoup indexes with ACCESS-INDEX. This is shown as a decimal value.

LEN
: is the length of the key used to select the collection element for recoup indexes with ACCESS-KEY. This is shown as a decimal value.

KEY
: is the key, displayed in hexadecimal character format, that is used to select the collection element for heterogeneous recoup indexes with ACCESS-KEY.

```
User:   ZBROW RECOUP DISPLAY INDEX001

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW1001I 10.56.43 RECOUP INDEX DISPLAY
        NAME - INDEX001
        PID  - 0002FC16 AF798883 E3D7C6C4 C2404040
               18036D0B 18036D0C 00000000 00000000
        DESC - PRIMARY_INDEX
        TYPE - HOMOGENEOUS    CONTROL - NONE
        TOKEN - C1C3E3F0F0F14040  TYPE-PID DISPL-40
        TOKEN - C1C3E3F0F0F24040  TYPE-PID DISPL-6F
        END OF DISPLAY
        BROW1014I 10.56.43 RECOUP INDEX DISPLAY SUCCESSFUL
```

The names of all recoup indexes defined in a data store are shown in the following example.

```
User:   ZBROW RECOUP DISPLAY ALL
System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TPFDB
        BROW1001I 10.56.43 RECOUP INDEX DISPLAY
          INDEX001
          INDEX002
          INDEX003
          INDEX004
          INDEX005
          IT00000
          IT00002
          IT00003
          IT00004
        END OF DISPLAY
        BROW1014I 10.56.43 RECOUP INDEX DISPLAY SUCCESSFUL
```

Recoup index INDEX001 is associated with the MYCOLLECT collection in the following example.

```
User:   ZBROW RECOUP LINK MYCOLLECT INDEX001

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1015I 10.56.43 RECOUP INDEX SUCCESSFULLY LINKED
```

The index entry corresponding to a key of TOKEN001 in recoup index INDEX001 is removed in the following example.

```
User:   ZBROW RECOUP REMOVE INDEX001 ETOK-TOKEN001

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1016I 10.56.43 RECOUP INDEX ENTRY SUCCESSFULLY REMOVED
```

The recoup index associated with the MYCOLLECT collection is disassociated with the collection in the following example.

```
User:   ZBROW RECOUP UNLINK MYCOLLECT

System: BROW0602I 10.56.43 BROWSER QUALIFIED FOR DSNAME TESTX.DS
        BROW1017I 10.56.43 RECOUP INDEX SUCCESSFULLY UNLINKED
```

Related Information

- See the “ZBROW QUALIFY–Qualify ZBROW for a Data Store” on page 154 for more information about how to specify to which data store the subsequent ZBROW command requests apply.
- See the “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See the *TPF Database Reference* for more information about the format of the file addresses embedded in collections.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZBUFC ALLOCATE–Specify Cache and Lock Allocation

Use this command to update the control information with new target values for either cache or lock space.

- For *cache* space, specify record slot allocation ratios for either 3880 record cache (RC) or 3990 record cache subsystem (RCS) control units in the complex.
- For *lock* space, specify the number of locks for the concurrency filter lock facility (CFLF).

Requirements and Restrictions

- This message only ***updates*** system control information with the desired allocation ratio values. Enter the ZBUFC ALLOCATE IMPLEMNT command to actually ***apply*** the pending changes for the specified record cache control units in the complex.
- In order to use the 3880 RC and 3990 RCS control units, you must partition subsystem storage so that it can use record and track (or buffer) slots in the cache storage.

Format

```
►►──ZBUFC ALLocate───────────────────────────────────────────────────────────►

            ┌──────────────┐
            ▼              │
►─┬──────┬─RC381-ratio───┴──────────────────────────────┬──────────────────►◄
  │      ├─RC1055-ratio─┤                                │
  │      ├─RC4096-ratio─┤                                │
  │      └─RCBUF-count──┘                                │
  │                         ┌───────────────┐            │
  │                         ▼               │            │
  ├─┬──────────┬──────────┬─RCS381-pp────┴──┬──────────────┬─┤
  │ └──INIT──┘            ├─RCS1055-pp──┤    └──Locks-num──┘ │
  │                       ├─RCS4096-pp──┤                    │
  │                       └─RCSBUF-pp───┘                    │
  └──Locks-num───────────────────────────────────────────────┘
```

RC381
: changes the record slot allocation for 381-byte records for 3880 RC control units.

RC1055
: changes the record slot allocation for 1055-byte records for 3880 RC control units.

RC4096
: changes the record slot allocation for 4096-byte (4-KB) records for 3880 RC control units.

ratio
: is a decimal ratio value from 1–255 for the record slot allocation.

RCBUF-*count*
changes the buffer counts for the 3880 record cache subsystem, where *count* is a decimal buffer count value from 1–16.

INIT
initializes the current and target allocation values of the record cache subsystem status table (SSST).

Note: You can specify this parameter only when you are prompted during restart.

RCS381
changes the record slot allocation for 381-byte records for 3990 RCS control units.

RCS1055
changes the record slot allocation for 1055-byte records for 3990 RCS control units.

RCS4096
changes the record slot allocation for 4096-byte (4-KB) records for 3990 RCS control units.

RCSBUF
changes the buffer counts for the 3990 record cache subsystem.

pp is a percentage from 0–95%.

Note: For the RCSBUF parameter, you can specify a percentage value from 5–25%.

Locks-*num*
changes the number of lock groups that are allocated, where *number* is a decimal number from 1–64.

Note: Locks are allocated in groups of 256. For example, LOCKS-1 allocates 256 locks and LOCKS-2 allocates 512 locks.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- If you omit any RC ratio value, the current ratio value retained by the TPF system is used for that record size.
- Do not define RC ratio values as 0 unless you intend to run the control unit in direct mode.
- For RCS allocations, the percentage values that you enter must total 100%.
- You can specify an equal sign (=) instead of the dash (-) for the parameters in this command.
- Processing for this command differs for the 3990 Model 3 and the IBM Enterprise Storage Server (ESS). See *TPF Database Reference* for more information about these processing differences.

Examples

The target allocation ratios used to partition 3880 RC control units are defined in the following example.

```
User:   ZBUFC ALLOC RC381-09 RC1055-08 RC4096-07 RCBUF-03

System: BUFC0037I 07.48.38 ALLOCATE - CACHE ALLOCATION PENDING
```

The target allocation ratios used to partition 3990 RCS control units are defined in the following example.

```
User:   ZBUFC ALLOC RCS381-25 RCS1055-30 RCS4096-20 RCSBUF-25

System: BUFC0037I 07.49.23 ALLOCATE - CACHE ALLOCATION PENDING
```

Related Information

See the following for more information about record caching:
- *TPF Database Reference*
- *3880 Storage Control Record Cache RPQ Introduction*
- *3880 Storage Control Record Cache RPQ Description*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC ALLOCATE DISPLAY–Display Cache Allocation Settings

Use this command to display the current and target allocation values for the following:

- 3880 record cache (RC) control unit
- 3990 record cache subsystem (RCS) control unit
- Concurrency filter lock facility (CFLF).

Requirements and Restrictions

None.

Format

```
                                 ┌ ──ALL─┐
►►──ZBUFC ALLocate DISplay───────┼───────┼──────────────────────────────────►◄
                                 ├ ──RC──┤
                                 └ ──RCS─┘
```

ALL
displays the current and target allocation values for both 3880 and 3990 record cache control units and for the concurrency filter lock facility (CFLF).

RC
displays the current and target allocation values for 3880 record cache (RC) control units.

RCS
displays the current and target allocation values for 3990 record cache subsystem (RCS) control units and the CFLF.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZBUFC HELP
ZBUFC ?

Examples

The following example displays the current and target allocation values for the 3880 RC control units.

```
User:   ZBUFC ALLOC DISP RC

System: BUFC0001I 07.54.24 RC CONTROL UNIT CACHE ALLOCATIONS

        CURRENT   RC381-  1,RC1055-  1,RC4096-  1,RCBUF- 1
        TARGET    RC381-  9,RC1055-  8,RC4096-  7,RCBUF- 3
```

The following example displays the current and target allocation values for the 3990 RCS control units and the CFLF.

ZBUFC ALLOCATE DISPLAY

```
User:   ZBUFC ALLOC DISP RCS

System: BUFC0002I 07.54.50 RCS CONTROL UNIT CACHE ALLOCATIONS

        CURRENT   RCS381-25,RCS1055-25,RCS4096-25,RCSBUF-25,LOCKS- 4
        TARGET    RCS381-25,RCS1055-30,RCS4096-20,RCSBUF-25,LOCKS- 4
```

The following example displays the current and target allocation values for the 3880 RC and 3990 RCS control units and the CFLF.

```
User:   ZBUFC ALLOC DISPLAY

System: BUFC0001I 07.51.38 RC CONTROL UNIT CACHE ALLOCATIONS

        CURRENT   RC381-  1,RC1055-  1,RC4096-  1,RCBUF- 1
        TARGET    RC381-  9,RC1055-  8,RC4096-  7,RCBUF- 3
        BUFC0002I 07.51.38 RCS CONTROL UNIT CACHE ALLOCATIONS

        CURRENT   RCS381-25,RCS1055-25,RCS4096-25,RCSBUF-25,LOCKS- 4
        TARGET    RCS381-25,RCS1055-30,RCS4096-20,RCSBUF-25,LOCKS- 4
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3880 Storage Control Record Cache RPQ Introduction*
- *3880 Storage Control Record Cache RPQ Description*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC ALLOCATE IMPLEMNT–Implement Pending Allocation Changes

Use this command to apply the allocation changes that were specified using the ZBUFC ALLOCATE command. This command applies the allocation changes to the control units that are not currently initialized with the new allocation values.

Requirements and Restrictions

If you are applying changes made for 3990 record cache control units, you must enter **ZBUFC FILE ALL** before you enter this command.

Format

```
                                           ┌─ ──TARget──┐
►►──ZBUFC ALLocate IMPlemnt── ──┬─RC──┬────┤            ├──────────────────►◄
                                ├─RCS─┤    └─ ──CURRent─┘
                                └─ALL─┘
```

RC
: applies the changes made for the 3880 (RC) record cache control units.

RCS
: applies the changes made for the 3990 record cache subsystem (RCS) control units and the concurrency filter lock facility (CFLF).

ALL
: applies the changes made for both the 3880 RC and 3990 RCS control units and for the CFLF.

TARget
: applies the target allocation values.

CURRent
: applies the current allocation values.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- If you enter this command while the ZMCPY function is running for a CFLF device, or while the ZAMOD function is placing a module online for a CFLF device, the ZMCPY or ZAMOD function ends with a CTL-0F5 dump.
- Processing for this command differs for the 3990 Model 3 and the IBM Enterprise Storage Server (ESS). See *TPF Database Reference* for more information about these processing differences.

Examples

The following example applies the target allocation values to all 3990 RCS control units in the complex.

ZBUFC ALLOCATE IMPLEMNT

```
User:   ZBUFC ALLOC IMP RCS

System: BUFC0061I 09.30.32 BUFC UTILITY OPERATION(S) STARTED TO RCS SSID 0003
        BUFC0039I 09.30.32 ALLOCATE - RCS CACHE ALLOCATION COMPLETE
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3880 Storage Control Record Cache RPQ Introduction*
- *3880 Storage Control Record Cache RPQ Description*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC ENABLE–Enable Caching Functions

Use this command to enable all the caching functions for one or all of the 3990 record cache subsystem (RCS) control units in the complex.

Requirements and Restrictions

This command is valid for only 3990 record cache subsystems.

Format

```
►►──ZBUFC ENAble── ──┬─device─┬──────────────────────────────────────►◄
                     └─ALL────┘
```

device
: is a 3990 record caching device address that is associated with a single record cache subsystem ID.

  Note: The caching functions are enabled for all the devices in the specified record cache subsystem.

ALL
: enables all the 3990 RCS control units in the complex.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- Enter the ZBUFC FILE command to disable the caching functions.
- Processing for this command differs for the 3990 Model 3 and the IBM Enterprise Storage Server (ESS). See *TPF Database Reference* for more information about these processing differences.

Examples

Caching functions for the 3990 RCS control unit associated with the specified address are enabled in the following example.

```
User:   ZBUFC ENABLE 04E7

System: BUFC0061I 13.49.46 BUFC UTILITY OPERATION(S) STARTED TO RCS SSID 0003
        BUFC0044I 13.49.46 ZBUFC ENABLE RCS SSID 0003 COMPLETE
```

Related Information

See the following for more information about record caching:
- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC FILE–Destage Modified Data from Cache to DASD

Use this command to:

- Destage any modified fast write data from cache to DASD to permit hardware reconfiguration or system shutdown.
- Disable cache and DASD fast write functions for one or all of the 3990 record cache subsystem (RCS) control units.

Requirements and Restrictions

This command is valid only for 3990 RCS control units.

Format

```
►►──ZBUFC FILe── ──┬─device─┬──────────────────────────────────────────►◄
                   └─ALL────┘
```

device
: is a 3990 record caching device address that is associated with a single record cache subsystem ID.

Note: The fast write data is destaged and the DASD and caching fast write functions are disabled for all the devices in the specified RCS.

ALL
: destages the fast write data and disables the DASD and caching fast write functions for all the 3990 RCS control units in the complex.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- Enter the ZBUFC ENABLE command to enable the DASD and caching fast write functions again.
- Even though this command disables the DASD and cache fast write functions, the cache is still active. Any fast write functions that are sent to the control unit are written to cache and the DASD surface.
- Processing for this command differs for the 3990 Model 3 and the IBM Enterprise Storage Server (ESS). See *TPF Database Reference* for more information about these processing differences.

Examples

In the following example, all the fast write data that is found in the 3990 RCS control unit and associated with the specified address is destaged, and additional fast write functions that are sent to this control unit are disabled.

```
User:   ZBUFC FILE 04E7

System: BUFC0061I 13.50.08 BUFC UTILITY OPERATION(S) STARTED TO RCS SSID 0003
        BUFC0042I 13.50.09 ZBUFC FILE RCS SSID 0003 COMPLETE
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC MAP–Map RCS SSID/Device Addresses

Use this command to display the 3990 record cache subsystem (RCS) ID that is associated with a given device address or the device range that is associated with a particular 3990 RCS ID.

Requirements and Restrictions

This command is valid only for 3990 RCS control units.

Format

```
►►──ZBUFC MAP── ──┬─DEVice-sda─┬───────────────────────────────────►◄
                  └─SSID-id────┘
```

DEVice-*sda*
: displays the 3990 RCS ID associated with a specific device, where *sda* is the 3- to 4-digit hexadecimal symbolic address of the device.

SSID-*id*
: displays device ranges associated with a specific subsystem ID, where *id* is the 1- to 4-digit hexadecimal RCS ID.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZBUFC HELP
ZBUFC ?

Examples

The following example displays the 3990 RCS ID that is associated with the control unit found at the specified address.

```
User:   ZBUFC MAP DEV-04E7

System: BUFC0012I 13.50.29 DEVICE 04E7 ATTACHED TO RCS SSID 0003
```

The following example displays the device range associated with the specified 3990 RCS ID.

```
User:   ZBUFC MAP SSID-0003

System: BUFC0013I 13.50.41 ATTACHED DEVICE RANGE(S) FOR RCS SSID 0003:
                           1E40-1E5F
                           END RCS-SSID MAP REPORT
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC PINNED DISCARD–Discard Pinned Data

Use this command to discard all pinned data associated with a device that is attached to a 3990 record cache subsystem (RCS).

Pinned data is data in the cache or in nonvolatile storage that could not be written to the DASD surface.

Requirements and Restrictions

- This command is valid only for 3990 RCSs.
- Use this command only for cache recovery. Incorrect use of this command can cause loss of data.

Format

```
►►──ZBUFC PINned DISCARD──┬───────┬── ──sda──────────────────────────►◄
                          └─ ──ALL─┘
```

ALL
discards all modified data in the cache for the device. If you do not specify the ALL parameter, only pinned data tracks are discarded.

sda
is the 3- to 4-digit hexadecimal symbolic address of a device.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- Enter the ZBUFC PINNED DISPLAY command to determined if there is any pinned data.
- Use this command with the ZBUFC STATUS command, which displays the current 3990 caching status, to help you determine what cache recovery actions are required.
- You can enter this command to an RCS device that is online or offline to the TPF system. If you enter this command to an offline device, the device address must be a valid DASD address that is defined to the TPF system.

Examples

In the following example, the pinned data is discarded for the 3990 RCS device that is found at the specified address.

```
User:   ZBUFC PIN DISCARD 04E7

System: BUFC0059I 13.50.54 CACHE RECOVERY REQUEST INITIATED ON DEVICE 04E7 RCS SSID 0003
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC PINNED DISPLAY–Display Pinned Data Report

Use this command to display the pinned data report for a device attached to a 3990 record cache subsystem (RCS).

Pinned data is data in the cache or in nonvolatile storage that could not be written to the DASD surface. The pinned data report includes a description of the type of pinned data that exists.

Requirements and Restrictions

This command is only valid for 3990 RCSs.

Format

```
                                   ┌─ ──00000000─┐
►►──ZBUFC PINned DISplay── ──sda───┼─────────────┼──────────────────────►◄
                                   └─ ──cccchhhh─┘
```

sda
: is the 3- to 4-digit hexadecimal symbolic address of a device.

cccchhhh
: is a an 8-digit hexadecimal pinned track starting address, where:

  cccc
  : is a 4-digit hexadecimal cylinder number.

  hhhh
  : is a 4-digit hexadecimal head number.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- As many as 8 track addresses are displayed with each report.
- Use this command with the ZBUFC STATUS command, which displays the current 3990 caching status, to help you determine what recovery actions are required.
- You can enter this command to an RCS device that is online or offline to the TPF system. If you enter this command to an offline device, the device address must be a valid DASD address that is defined to the TPF system.

Examples

The following example displays the pinned data report for the 3990 RCS device found at the specified address.

Notice that only 8 track addresses are displayed at a time. Enter the ZBUFC PINNED DISPLAY again and specify the address marked with 2 asterisks (**) to display any pinned data tracks that exist beyond the last address.

ZBUFC PINNED DISPLAY

```
User:   ZBUFC PIN DIS 04E7

System: BUFC0014I 13.51.24 PINNED TRACK REPORT FOR DEVICE 04E7 ON RCS SSID 0003
                           CCCCHHHH          PINNED DATA TYPE

                           00010002          NON-RETRIABLE
                           00010007          NON-RETRIABLE
                           0001000E          PINNED IN NVS
                           00A20001          RETRIABLE
                           00A20002          RETRIABLE
                           01BF000C          CACHE COPY DEFECTIVE
                           01BF000D          CACHE COPY DEFECTIVE
                           01BF000E**        CACHE COPY DEFECTIVE

                           MORE PINNED DATA TRACKS EXIST
                           USE (**) CCCCHHHH FOR NEXT REQUEST
```

The following example displays the pinned data report for the 3990 RCS device found at the specified address, beginning with track address 00010001.

```
User:   ZBUFC PINNED DISPLAY 04E7 00010001

System: BUFC0014I 13.51.42 PINNED TRACK REPORT FOR DEVICE 04E7 ON RCS SSID 0003

                           CCCCHHHH          PINNED DATA TYPE
                           00010001          RETRIABLE
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC SETCACHE–Set Cache Operating Modes

Use this command to change the cache resource states for the 3990 record cache subsystem (RCS) to perform cache error recovery options.

Requirements and Restrictions

- This command is valid for only 3990 RCSs.
- Use this command for cache recovery only. Incorrect use of this command can cause loss of data.

Format

```
►►──ZBUFC SETcache── ──sda── ──┬─SSCACHE──┬── ──┬─ON─────┬──────────────────►◄
                               ├─CFW──────┤     ├─OFF────┤
                               ├─NVS──────┤     ├─FORCE──┤
                               ├─DEVCACHE─┤     └─REINIT─┘
                               ├─DEVDFW───┤
                               └─DEVRECM──┘
```

sda
: is the 3- to 4-digit hexadecimal symbolic address of a device.

SSCACHE
: sets the SSCACHE function, where:

  ON
  : Allows the 3990 subsystem storage (volatile cache) to be logically brought online for normal caching operations.

  OFF
  : Allows the 3990 subsystem storage (volatile cache) to be logically taken offline, preventing normal caching operations.

  FORCE
  : Allows the 3990 subsystem storage (volatile cache) to be logically taken offline from a pending offline state.

  REINIT
  : Resets the 3990 cache and nonvolatile storage operating states to a default state and reinitializes all status tracks. Use this option with caution because all modified cache data is invalidated, possibly resulting in data loss from the volatile and nonvolatile storage.

CFW
: sets the CFW function, where:

  ON
  : Allows the 3990 subsystem storage cache fast write function to be logically brought online for the subsystem.

  OFF
  : Allows the 3990 subsystem storage cache fast write function to be logically taken offline for the subsystem.

NVS
: sets the NVS function, where:

ON
Allows the 3990 subsystem nonvolatile storage to be logically brought online for the subsystem.

OFF
Allows the 3990 subsystem nonvolatile storage to be logically taken offline for the subsystem.

DEVCACHE
sets the DEVCACHE function, where:

ON
Allows caching to be started for the specified RCS device address.

OFF
Allows caching to be stopped for the specified RCS device address.

DEVDFW
sets the DEVDFW function, where:

ON
Allows DASD fast write to be started for the specified RCS device address.

OFF
Allows DASD fast write to be stopped for the specified RCS device address.

FORCE
Allows DASD fast write to be stopped by force from a pending offline state for the specified RCS device address.

DEVRECM
sets the DEVRECM function, where:

ON
Allows record mode to be started for the specified RCS device address.

OFF
Allows record mode to be stopped for the specified RCS device address.

Note: Some options for the ZBUFC SETCACHE function may be rejected by the 3990 RCS control unit if they are not compatible with the existing cache status.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- Use this command with the ZBUFC STATUS command, which displays the current 3990 caching status, to help you determine what cache recovery actions are required.
- You can enter this command to an RCS device that is online or offline to the TPF system. If you enter this command to an offline device, the device address must be a valid DASD address that defined to the TPF system.
- Processing for this command differs for the 3990 Model 3 and the IBM Enterprise Storage Server (ESS). See *TPF Database Reference* for more information about these processing differences.

Examples

The SSCACHE function is turned on in the following example.

```
User:   ZBUFC SETC 04E7 SSCACHE ON

System: BUFC0059I 13.51.53 CACHE RECOVERY REQUEST INITIATED ON DEVICE 04E7 RCS SSID 0003
```

The CFW function is turned off in the following example.

```
User:   ZBUFC SETCACHE 04E7 CFW OFF

System: BUFC0059I 13.52.24 CACHE RECOVERY REQUEST INITIATED ON DEVICE 1E53 RCS SSID 0003
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC STATUS–Display Caching Status

Use this command to the display current caching status information for a 3880 record cache (RC) or 3990 record cache subsystem (RCS) control unit. The display shows the device operating modes, storage and interface status, and the storage size allocated to each TPF block size.

Requirements and Restrictions

For devices that are attached to concurrency filter lock facility (CFLF) control units, this command resets the high watermark count of locks that is used by data collection. Therefore, do not enter this command while the data collection function is running.

Format

```
►►──ZBUFC STAtus── ──┬─sda─┬──────────────────────────────────────────────►◄
                     └─RCS─┘
```

sda
: is the 3- to 4-digit hexadecimal symbolic address of a record caching device.

RCS
: displays global 3990 RCS information from the internal control information that is maintained by the TPF system.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZBUFC HELP
ZBUFC ?

Examples

The current record caching status information for the specified 3990 RCS control unit is displayed in the following example.

```
User:   ZBUFC STATUS 04E7

System: BUFC0004I 13.52.46 STATUS FOR DEVICE-1E53 RCS SSID-0003

        DEVICE COUNT -  32     STATISTICS COUNT/DEVICE -   4
          CONFIGURED SS STORAGE   - 02000000
          AVAILABLE SS STORAGE    - 01DC4000
          PINNED SS STORAGE       - 00000000
          OFFLINE SS STORAGE      - 00000000
          CONFIGURED NVS STORAGE  - 00400000
          PINNED NVS STORAGE      - 00000000

        CACHE ALLOCATION:  (RECORD SLOT SIZES 1-3)
          SIZE    - TRACK     381    1055     4096
          RATIO   -    25      25      25       25
          LOCKS   -     4

        SUBSYSTEM STATUS:
          SUBSYSTEM CACHING       - ACTIVE
          SD CACHING CONDITIONS   - NONE
          CACHE FAST WRITE        - DISABLED
          CACHE FAST WRITE ID     - 0008
          NONVOLATILE STORAGE     - DEACTIVATED-HOST REQUEST
          NVS CACHING CONDITIONS  - NONE
          FAST WRITE SUSPENDED    - NO
          SS DISABLE REQUESTED    - YES
          QUEUE THRESHOLD ACTIVE  - YES
        DEVICE STATUS:
          DEVICE CACHING          - ACTIVE
          DEVICE DASD FAST WRITE  - ACTIVE
          PINNED DATA             - NONE
          RECORD MODE             - ALLOWED
          DATA LOSS EXPOSURE      - NO
```

Global 3990 RCS information that is maintained by the TPF system is displayed in the following example.

```
User:   ZBUFC STATUS RCS

System: BUFC0005I 13.53.25 RCS GLOBAL STATUS DISPLAY

        ACTIVE SUBSYSTEM COUNT    -     1
        I/O QUEUE THRESHOLD VALUE - 16384
        COMPLEX DISABLE REQUESTED - YES
        RETENTIVE OVERRIDE ACTIVE - NO
        ACTIVE 'BUFC' UTILITY     - NONE
```

Related Information

See the following for more information about record caching:

- *TPF Database Reference*
- *3880 Storage Control Record Cache RPQ Introduction*
- *3880 Storage Control Record Cache RPQ Description*
- *3990 Storage Control Planning, Installation, and Storage Administration*
- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZBUFC THRESHLD–Display or Change I/O Queue Threshold Value

Use this command to display or change the current I/O queue threshold value for all 3990 record cache subsystem (RCS) control units.

Requirements and Restrictions

This command is valid for only 3990 RCSs.

Format

```
►►──ZBUFC THReshld──┬─────────┬──────────────────────────────────►◄
                    └──value──┘
```

value
is a decimal value, from 1–32 767, that represents the maximum queue depth for each device.

This value is used to calculate the maximum overall I/O queue threshold value that the 3990 RCS can tolerate during degraded caching operations.

If you do not specify a value, the device threshold value that is currently retained in the RCS status table (SSST) is displayed.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZBUFC HELP
  ZBUFC ?
- If the overall subsystem queue threshold is exceeded, a user exit starts with I/O queue count information to allow you to specify various processing actions.

Examples

The current I/O queue threshold value for the 3990 RCS devices is displayed in the following example.

```
User:   ZBUFC THRES

System: BUFC0011I 08.06.02 RCS DEVICE I/O QUEUE THRESHOLD VALUE - 16384
```

The current I/O queue threshold value for the 3990 RCS devices is changed in the following example.

```
User:   ZBUFC THRES 1234

System: BUFC0011I 08.06.41 RCS DEVICE I/O QUEUE THRESHOLD VALUE -  1234
```

Related Information

See the following for more information about record caching:
- *TPF Database Reference*
- *3990 Storage Control Planning, Installation, and Storage Administration*

- *3990 Storage Control Operations and Recovery Guide*
- *3990 Transaction Processing Facility Support RPQs*.

ZCACH–Manage Logical Record Caches

Use this command to manage logical record caches.

Requirements and Restrictions

You can enter this command only from the basic subsystem (BSS) when the following parameters are specified:

- CF DISABLE
- CF ENABLE
- DELETE
- FLUSH
- MODIFY
- REMOVE.

Format

```
►►──ZCACH── ──┬─MODify── ──cache── ──SIZE entrysize──┬──────────────────►◄
              ├─DISPlay── ──┬─CTL──┬─────────┬──┬───┤
              │             │      └─ ──FILE─┘  │   │
              │             ├─ATTributes── ──cache─┤ │
              │             ├─SYNonym── ──cache────┤ │
              │             └─COunts── ──cache─────┘ │
              ├─CF ENABLE────────────────────────────┤
              ├─CF DISABLE───────────────────────────┤
              ├─REMove── ──cache─────────────────────┤
              ├─FLush── ──cache──────────────────────┤
              └─DELete── ──cache─────────────────────┘
```

MODIFY
: sets the number of entries defined for the logical record cache in the processor unique logical record cache control record. The number of entries specified here overrides the number of entries specified on the `newCache` function call that is issued by the application to create the logical record cache. The new number of entries is not active until the specified logical record cache is created.

cache
: is the 4- to 12- character alphanumeric name of the target logical record cache. The first character of the logical record cache name must be an alphabetic character.

SIZE *entrysize*
: specifies the number of entries defined for the logical record cache, where *entrysize* is a decimal number, from 0 to 999 999 999.

DISPlay
: displays information about the logical record cache.

CTL
: displays the names of the processor's current logical record caches and their addresses from main storage.

FILE
displays the names of the current logical record caches and their assigned sizes as defined in the processor unique cache control record.

ATTributes
displays the attributes of the specified logical record cache and the associated CF if a CF is in use.

SYNonym
displays the synonym chain length information for the specified logical record cache.

COunts
displays the current data collection counters for a specific logical record cache.

CF ENABLE
enables the processor shared caches to use a CF for cache synchronization.

CF DISABLE
disables the processor shared caches from using a CF for cache synchronization.

REMove
removes the cache definition for the specified logical record cache from the processor unique cache control record. This action does not affect the logical record cache that is currently active.

FLush
clears all entries from the processor's logical record cache. There is no notification sent to other processors.

DELete
deletes the specified logical record cache from the processor.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZCACH HELP
ZCACH ?

Examples

The following information is displayed in the examples:

CALLS
The number of `readCacheEntry` calls.

CASTOUTS
The number of times a cache tosses out a valid entry to make room for another entry.

DUPLICATE HASH
The number of times an entry was not added to a cache because its 16 byte name was the same as an entry already in the cache.

INVALIDATED
The number of times a cache entry was invalidated by another processor.

MISSED
The number of `readCacheEntry` calls that did not find a valid entry.

UPDATES
The number of times a `readCacheEntry` call caused a cross invalidate to the CF.

The following example shows the names of the current logical record caches and their addresses.

```
User:    ZCACH DISP CTL

System:  CACH0002I 19.29.52 CACHE CONTROL AREA DISPLAY
         NAME TPF_FS_DIR     ,ADDR 05073000
         NAME TPF_FS_INODE   ,ADDR 05287000
         DISPLAY COMPLETE
```

The following example shows the names of the current logical record caches and their assigned sizes as defined in the processor unique cache control record.

```
User:    ZCACH DISP CTL FILE

System:  CACH0026I 09.41.56 CACHE CONTROL RECORD DISPLAY
         NAME  CACHE12456, ENTRY COUNT  000012456
         DISPLAY COMPLETE
```

The following example shows the attributes for logical record cache TPF_FS_DIR, which is using CF CFLOCK2 for cache synchronization.

```
User:    ZCACH DISP ATT TPF_FS_DIR

System:  CACH0004I 12.32.15 CACHE ATTRIBUTE DISPLAY
         NAME   TPF_FS_DIR         ,ADDR 050B9000,    CF CFLOCK2
         NUMBER HASH ENTRIES  97   NUMBER ENTRIES       0
         PRIMARY KEY SIZE    256   SECONDARY KEY SIZE   4
         ENTRY SIZE          328   DATA LENGTH         32
         CAST OUT TIME        60
         DISPLAY COMPLETE
```

The following example shows the attributes for logical record cache TPF_FC_DIR. This particular logical record cache does not use a CF for cache synchronization.

```
User:    ZCACH DISP ATT TPF_FC_DIR

System:  CACH0004I 12.32.15 CACHE ATTRIBUTE DISPLAY
         NAME   TPF_FC_DIR         ,ADDR   050B9000
         NUMBER HASH ENTRIES  97   NUMBER ENTRIES       0
         PRIMARY KEY SIZE    256   SECONDARY KEY SIZE   4
         ENTRY SIZE          328   DATA LENGTH         32
         CAST OUT TIME        60
         DISPLAY COMPLETE
```

The following example shows synonym chain length information for logical record cache TPF_FS_DIR.

```
User:    ZCACH DISP SYN TPF_FS_DIR

System:  CACH0006I 12.32.15 CACHE SYNONYM DISPLAY
         NAME   TPF_FS_DIR         ,ADDR   050B9000
         HASH ENTRIES  97   LONGEST SYNONYM CHAIN 0
         HASH CHAINS WITH 0 ENTRIES 97
         HASH CHAINS WITH 1 ENTRIES 0
         HASH CHAINS WITH 2 ENTRIES 0
         HASH CHAINS WITH 3 ENTRIES 0
         HASH CHAINS WITH 4 ENTRIES 0
         HASH CHAINS WITH 5 ENTRIES 0
         DISPLAY COMPLETE
```

The following example shows the current data collection counters for logical record cache TPF_FS_DIR.

```
User:    ZCACH DISP CO TPF_FS_DIR

System:  CACH0005I 12.32.15 CACHE DATA COLLECTION COUNTER DISPLAY
         NAME   TPF_FS_DIR ,ADDR   093C0000
         NUMBER ENTRIES     100     CAST OUT TIME  60
         CALLS               0      CASTOUTS        0
         UPDATES             0      INVALIDATED     0
         DUPLICATE HASH REFUSALS      0_
         DISPLAY COMPLETE
```

The following example shows processor shared caches being enabled to use the CF for cache synchronization.

```
User:    ZCACH CF ENABLE

System:  CACH0019I 12.32.15 CACHE SUPPORT ENABLED TO USE CF
```

Related Information

- See “ZCFCH–Manage Coupling Facility Cache Structures” on page 194 for information about managing CF cache structures.
- See *TPF Database Reference* for more information about logical record cache and CF cache support.
- See *TPF C/C++ Language Support User’s Guide* for more information about the `newCache` function.

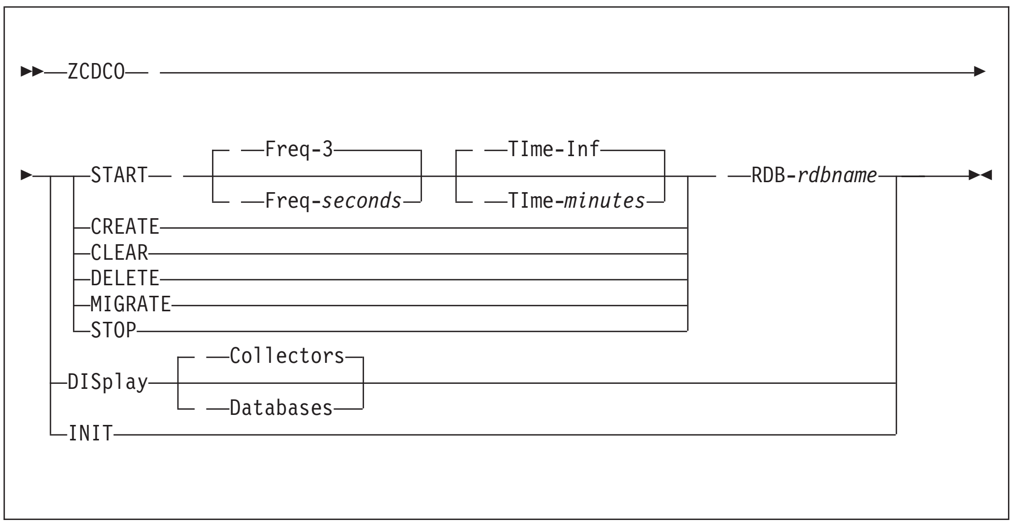
ZCDCO–Managing Continuous Data Collection

Use this command to manage TPF Application Requester (TPFAR) continuous data collection (CDC).

Requirements and Restrictions

- The TPFAR feature must be installed on your TPF system.
- You can start CDC when the TPF system is in 1052 state or above; however, data will not be collected unless the TPF system is in CRAS state or above because communications are not active.
- You can specify the CREATE, CLEAR, DELETE, or MIGRATE parameter only in CRAS state or above.
- You cannot enter the ZCDCO command before the TPF file system is initialized. Use the ZFINT command to initialize the TPF file system.

Format



START
starts TPFAR CDC.

Freq
specifies how often a TPFAR CDC record is collected, where:

3 indicates that the record is collected every 3 seconds.

seconds
is the number of seconds in the range 1–99.

TIme
specifies how long a TPFAR CDC record is generated, where:

Inf indicates that the record is generated until TPFAR CDC is stopped.

minutes
is the number of minutes in the range 1–1440.

RDB-*rdbname*
specifies the name of a remote relational database, where *rdbname* is the 1- to 18-character alphanumeric name.

Note: Ensure that you entered the ZSQLD command previously to define the database specified by the RDB parameter.

CREATE
creates the TPFAR CDC table on the specified remote relational database and adds an entry to the remote relational database list.

CLEAR
clears all entries in the TPFAR CDC table on the specified remote relational database.

DELETE
deletes the TPFAR CDC table on the specified remote relational database.

MIGRATE
migrates the TPFAR CDC table on the specified remote relational database to the current level needed to run CDC.

STOP
stops TPFAR CDC on the specified remote relational database.

DISplay
displays TPFAR CDC information, where:

COLLECTORS
displays the items that have been started and are currently running.

DATABASES
displays databases and tables that have been created or defined.

INIT
clears the list of all the remote relational databases on the TPF system.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCDCO HELP
  ZCDCO ?
- Enter the ZSTIM A command to maintain TPFAR CDC over a system re-IPL by automatically restarting the collector in NORM state.
- To enhance performance, run CDC with *hotcons* (hot conversations or hot connections). Enter the ZSQLD command with the MAXHC parameter specified to set the number of hotcons.
- The name of the table created on the remote relational database is SYSTPF.TPF_DATA.

Examples

In the following example, the TPFAR CDC table is created.

```
User:   ZCDCO CREATE RDB-TPFDB

System: CDCO0007I 10.01.37 RDB-TPFDB -CREATED CONTINUOUS DATA COLLECTION
                           TABLE ON THE REMOTE DATABASE
```

In the following example, all previously created TPFAR CDC tables are displayed.

```
User:   ZCDCO DISPLAY DATABASES

System: CDCO0020I 10.01.37 DISPLAYING CREATED CONTINUOUS DATA COLLECTION TABLES
        DB                  TABLE
        ----------------    ----------
        TPFDB               IBM_TPF_DATA
        END OF DISPLAY
```

In the following example, TPFAR CDC is started for remote relational database TPFDB.

```
User:   ZCDCO START RDB-TPFDB FREQ-10

System: CDCO0002I 10.01.37 RDB-TPFDB -CONTINUOUS DATA COLLECTION HAS BEEN STARTED
```

In the following example, all CDC collectors that are currently running are displayed.

```
User:   ZCDCO DISPLAY COLLECTORS

System: CDCO0016I 10.01.37 DISPLAYING RUNNING CONTINUOUS DATA COLLECTORS
        DB      TABLE           FREQ    RUN TILL    LAST RUN    NEXT RUN
        ----    ----------      ----    --------    --------    --------
        TPFDB   IBM_TPF_DATA      5     10:02:29    10:01:34    10:01:39
        END OF DISPLAY
```

In the following example, TPFAR CDC is stopped for remote relational database TPFDB.

```
User:   ZCDCO STOP RDB-TPFDB

System: CDCO0003I 10.01.37 RDB-TPFDB - CONTINUOUS DATA COLLECTION HAS BEEN STOPPED
```

In the following example, TPFAR CDC records are cleared from remote relational database TPFDB.

```
User:   ZCDCO CLEAR RDB-TPFDB

System: CDCO0006I 10.01.37 RDB-TPFDB - CLEARED ALL CONTINUOUS DATA COLLECTION
                           RECORDS FROM THE REMOTE DATABASE
```

In the following example, a TPFAR CDC table is deleted from remote relational database TPFDB.

```
User:   ZCDCO DELETE RDB-TPFDB

System: CDCO0019I 10.01.37 RDB-TPFDB -CONTINUOUS DATA COLLECTION TABLE DELETED
                           FROM THE REMOTE DATABASE
```

In the following example, the list of all the remote relational databases on the TPF system is cleared.

```
User:   ZCDCO INIT

System: CDCO0012I 10.01.37 -THE REMOTE DATABASE LIST HAS BEEN INITIALIZED
```

Related Information

- See the *TPF Application Requester User's Guide* for more information about the TPFAR feature.
- See *TPF System Performance and Measurement Reference* for more information about TPFAR CDC.

ZCFCH–Manage Coupling Facility Cache Structures

Use this command to manage coupling facility (CF) cache structures.

Requirements and Restrictions

This command must be entered from the basic subsystem (BSS).

Format

```
►►──ZCFCH── ──┬─DISPlay── ──┬─COunts── ──cache─┬─────┬──────────────────────────►◄
              │             └─FILE─────────────┘     │
              ├─MODify── ──cache── ──SIZE entrysize─┤
              └─REMove── ──cache────────────────────┘
```

DISPlay
: displays information about the CF cache structure.

COunts
: displays the current storage class counters for the CF cache structure.

cache
: is the 4- to 12- character alphanumeric name of the target CF cache structure. The first character of the CF cache structure name must be an alphabetic character.

FILE
: displays the contents of the processor shared CF cache control record.

MODify
: sets the number of entries defined for the CF cache structure in the processor shared CF cache control record. The number of entries specified overrides the number of entries specified on the `newCache` function call that is issued by the application to create the CF cache structure. The change is not active until the next time theCF cache structure is created on the CF.

SIZE *entrysize*
: specifies the number of entries defined for the CF cache structure where *entrysize* is a decimal number from 0 to 999 999 999.

REMove
: allows you to remove the cache definition for the specified CF cache structure from the processor shared CF cache control record. This action does not affect the CF cache structure that is currently active.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCFCH HELP
ZCFCH ?

Examples

The following example shows information about the processor shared CF cache control record.

```
User:    ZCFCH DISP FILE

System:  CFCH0002I 19.29.52 CF CACHE CONTROL RECORD DISPLAY
         NAME HISNHER       SIZE   000000200
         NAME HERNHIS       SIZE   000000300
         DISPLAY COMPLETE
```

The following example shows information about the current storage class counters for the CF cache structure.

```
User:    ZCFCH DISP CO TPF_FS_INODE

System:  CFCH0005I 15.15.48 CF STORAGE CLASS 1 COUNTER DISPLAY
         NAME TPF_FS_INODE
         DIRECTORY HIT               0000000008
         READ MISS, ASSIGNED         0000000064
         DIRECTORY                   0000000064
         COUNTERS WITH ZERO WERE NOT DISPLAYED
         DISPLAY COMPLETE
```

The following example changes the number of entries for CF cache structure HISNHER to 300.

```
User:    ZCFCH MODIFY HISNHER SIZE 300

SYSTEM:  CFCH0017I 12.20.22 CACHE HISNHER    SUCCESSFULLY MODIFIED
```

Related Information

- See “ZCACH–Manage Logical Record Caches” on page 186 for more information about managing logical record caches.
- See *TPF Database Reference* for more information about logical record cache and CF cache support.
- See *TPF C/C++ Language Support User’s Guide* for more information about the `newCache` function.

ZCFLK ADD–Add a Coupling Facility to the Coupling Facility Locking Configuration

Use this command to add a coupling facility (CF) to the CF locking configuration, which then enables the CF for use as an external locking facility (XLF) so new locks can be stored on it. Lock residency is recalculated on a module-by-module basis for all modules that are currently running with CF locking to include this CF.

Requirements and Restrictions

- You can enter this command only after the CF you want added to the CF locking configuration has been added to the processor configuration. Enter the ZMCFT ADD command to do so.
- Using your current lockspace requirements, determine a reasonable size, in 4-KB blocks, for the lockspace structure in the CF before entering this command. See *TPF Database Reference* and your data collection information to determine a reasonable size based on your current lockspace requirements.

Format

```
►►──ZCFLK ADD── ──cfname── ──SIZE── ──num4k──────────────────────────────────►◄
```

cfname
: is the 5- to 8-character alphanumeric name of the CF you want to add to the CF locking configuration. The first character of the CF name must be an alphabetic character.

SIZE *num4k*
: specifies the size of the lockspace structure in the CF, where *num4k* is a 4- to 8-digit decimal number of 4-KB blocks. You can specify a minimum of one thousand 4-KB blocks.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCFLK HELP
ZCFLK ?

Examples

The following example shows a CF named LOCKCF01 being added to the CF locking configuration.

```
User:    ZCFLK ADD LOCKCF01 SIZE 1000

System:  CFLK0002I 15.54.26 CFLO - COUPLING FACILITY LOCKCF01 WAS ADDED TO THE
                             LOCKING CONFIGURATION
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support and lockspace requirements.

- See “ZMCFT ADD–Add a Coupling Facility” on page 754 for more information about the ZMCFT ADD command.

ZCFLK DELETE–Remove a Coupling Facility from the Coupling Facility Locking Configuration

Use this command to remove a coupling facility (CF) from the CF locking configuration. All locks stored on this CF are redistributed automatically among the remaining CFs in the CF locking configuration on a module-by-module basis. As a result, locks can no longer be stored on this CF.

Requirements and Restrictions

You can enter this command only when the CF you want to remove from the CF locking configuration is ***not*** the last CF defined in the CF locking configuration.

If the CF is the last one defined in the CF locking configuration and at least one module is using CFs for locking, the lock residency for all affected modules must be migrated back to the locking control units (CUs). You can do so by using the ZCFLK MIGRATE command. If you are migrating an online module, the duplicate module is also migrated. You can then enter the ZCFLK DELETE command to remove the CF from the locking configuration.

Format

```
►►──ZCFLK DELete── ──cfname──────────────────────────────────────────────►◄
```

cfname
: is the 5- to 8-character alphanumeric name of the CF you want to remove from the CF locking configuration. The first character of the CF name must be an alphabetic character.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCFLK HELP
ZCFLK ?

Examples

The following example shows the CF named LOCKCF03 being deleted from the CF locking configuration.

```
User:    ZCFLK DELETE LOCKCF03

System:  CFLK0003I 12.40.21 CFLDEL - COUPLING FACILITY LOCKCF03 WAS DELETED FROM
                             THE LOCKING CONFIGURATION
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See "ZCFLK MIGRATE–Migrate Lock Residency in the Coupling Facility Locking Configuration" on page 202 for more information about the ZCFLK MIGRATE command.

ZCFLK DISPLAY–Display the Coupling Facility Locking Configuration

Use this command to display information about the coupling facility (CF) locking configuration.

Requirements and Restrictions

The counts displayed only show the results for the subsystem from which the command was entered.

Format

```
►►─ZCFLK DISPlay─ ──┬─CF─ ──┬─cfname─┬──┬──────────────────────────────────►◄
                    │       └─ALL────┘  │
                    └─MOD─ ─modnum──────┘
```

CF *cfname*
displays information about a specific CF, where *cfname* is the 5- to 8-character alphanumeric name of the CF whose status you want to display. The first character of the CF name must be an alphabetic character.

ALL
displays information for all CFs in the CF locking configuration.

MOD *modnum*
displays information about a specific module, where *modnum* is the 1- to 4-digit hexadecimal symbolic module number.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCFLK HELP
  ZCFLK ?
- To display the status of one or more CFs in the processor configuration, enter the ZMCFT DISPLAY command.
- To display the module file status table (MFST) information for one or more modules assigned to the basic subsystem (BSS) or a subsystem in a multiple database function (MDBF) environment, enter the ZDMFS command. The module status (or status indicator) in the informational display refers to the CF, not to the module.

Examples

The following information is displayed in the examples:

CFNAME
is the 5- to 8-character alphanumeric name of the CF whose status you are displaying.

CFLT_ADDRESS
is the address of the CF entry in the CF locking table (CFLT).

MODS_ASSIGNED
is the number of modules actively running with locking on the CF

MOD
  is the 1- to 4-digit symbolic module number.

CF_ASSIGNED
  is the name of the CF assigned to the module.

ACTIVE
  indicates whether the module is or is not using a CF for locking; either YES or NO.

The following example shows the ZCFLK DISPLAY command being used to display information about a CF named LOCKCF03 in the CF locking configuration.

```
User:    ZCFLK DISPLAY CF LOCKCF03

System:  CFLK0015I 03.22.54 CFLD - ZCFLK DISPLAY STARTS

         CFNAME     CFLT_ADDRESS MODS_ASSIGNED
         LOCKCF03     0403AB00        16

         END OF ZCFLK DISPLAY
```

The following example shows the ZCFLK DISPLAY command being used to display information about symbolic module 047 in the CF locking configuration.

```
User:    ZCFLK DISPLAY MOD 047

System:  CFLK0021I 03.22.54 CFLD - ZCFLK DISPLAY STARTS

         MOD CF_ASSIGNED ACTIVE
         047  LOCKCF01   YES

         END OF ZCFLK DISPLAY
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See “ZMCFT DISPLAY–Display Coupling Facility Status” on page 757 for more information about the ZMCFT DISPLAY command.
- See “ZDMFS–Display Module File Status” on page 342 for more information about the ZDMFS command.

ZCFLK INITIALIZE–Initialize the Coupling Facility Locking Configuration

Use this command to initialize the coupling facility (CF) locking configuration.

Requirements and Restrictions

- Entering this command while using CF record lock support may corrupt the database. The ZCFLK INITIALIZE command should only be used to initially set up CF record lock support fixed file records with the correct data or when a CF locking error occurs. The ZCFLK INITIALIZE command only initializes the configuration on file.

  Be sure to perform an initial program load (IPL) of the TPF system after entering the ZCFLK INITIALIZE command to initialize the configuration in main storage. If CFs are being used for CF locking at the time this command is entered, you should change the lock residency of the modules from CFs to concurrency filter lock facility (CFLF) locking control units (CUs) by entering the ZCFLK MIGRATE command.
- The TPF system must be in restart and be the first processor to IPL in the loosely coupled complex ***before*** you can enter this command.

Format

```
►►──ZCFLK INITialize───────────────────────────────────────────────────►◄
```

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCFLK HELP
ZCFLK ?

Examples

The following example shows the CF locking configuration being initialized.

```
User:    ZCFLK INIT

System:  CFLK0028I 12.56.00 CFLK - THE COUPLING FACILITY LOCKING DATABASE
                             HAS BEEN INITIALIZED SUCCESSFULLY
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See “ZCFLK MIGRATE–Migrate Lock Residency in the Coupling Facility Locking Configuration” on page 202 for more information about the ZCFLK MIGRATE command and migrating the lock residency in the CF locking configuration.

ZCFLK MIGRATE–Migrate Lock Residency in the Coupling Facility Locking Configuration

Use this command to change the lock residency of a module from a concurrency filter lock facility (CFLF) locking control unit (CU) to a coupling facility (CF) , or from a CF to a CFLF locking CU.

Requirements and Restrictions

- At least one CF must be added to the CF locking configuration by using the ZCFLK ADD command before entering this command.
- If you are migrating an online module, the duplicate module is also migrated to ensure that the locks for a given module reside on only a CFLF locking CU or a CF.
- The range of module numbers specified (either one module number or multiple module numbers) must be real-time symbolic module numbers for the subsystem from which you enter this command. The *range* begins with the first module number specified and continues consecutively.
- You cannot specify general file (GF) relative module numbers because locks do not exist for a GF module.
- If you were running on a TPF system without locking in forced uniprocessor mode before issuing the ZCFLK MIGRATE command to migrate to CF record lock support, you must perform an initial program load (IPL) of the TPF system in order to use locking.

Format

```
►►──ZCFLK MIGrate── ──┬─modnum────────┬── ──TO── ──┬─CF─┬──────────────────────►◄
                      ├─modnum.modnum─┤            └─CU─┘
                      ├─ONLine────────┤
                      └─GDS───────────┘
```

modnum
: is the 1- to 4-digit symbolic module number.

modnum.modnum
: specifies a range of module numbers for which the corresponding locking facilities are migrated. The *range* begins with the first module number specified and continues consecutively.

ONLine
: specifies that the lock residency for all online modules is directed to a CFLF locking CU or a CF.

GDS
: specifies that the lock residency for general data set (GDS) modules is directed to a CFLF locking CU or a CF.

CF
: specifies that the lock residency for the specified module is directed to a CF.

CU
specifies that the lock residency for the specified module is migrated to a CFLF locking CU.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCFLK HELP
ZCFLK ?

Examples

The following example shows the migration of lock residency for a range of modules (047 to 04A) to a CF.

```
User:    ZCFLK MIGRATE 047.04A TO CF

System:  CFLK0004I 23.56.00 CFLW - ZCFLK MIGRATE PROCESSING COMPLETE
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See "ZCFLK ADD–Add a Coupling Facility to the Coupling Facility Locking Configuration" on page 196 for more information about the ZCFLK ADD command.

ZCHCH–Chain Chase Utility

Use this command to obtain a list of file pool addresses chained from a specific location in a specific record, and display each file pool address or only the last address in the chain.

Requirements and Restrictions

None.

Format

```
►►──ZCHCH── ──fileaddr── ──┬─F── ──num───────────────────┬─ ──HDR-─┬─4─┬──────────►
                           ├─B── ──num───────────────────┤         └─8─┘
                           └─U── ──num── ──displacement──┘

►──┬───────┬─────────────────────────────────────────────────────────────────►◄
   └─ ──L──┘
```

fileaddr
: is the 8- or 16-digit hexadecimal file pool address of the record from which the chain chase begins.

F chases the standard forward chains.

B chases the standard backward chains.

U chases the chain address at the location specified for displacement.

num
: is the maximum number (1–999) of chained records to be chased.

displacement
: is the 3-byte hexadecimal displacement (location) in the chain chase at which the file pool address of the next record in that chain chase is found.

HDR
: specifies the type of standard header. Specify one of the following:

  4 specifies that the records have a 4-byte standard header. If you specify this value, you can specify a 4-byte file address or an 8-byte file address in 4x4 format.

  8 specifies that the records have an 8-byte standard header. If you specify this value, you can specify an 8-byte FARF6 address or an 8-byte file address in 4x4 format. Do ***not*** specify a 4-byte file address.

L displays details about only the last record in the chain.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCHCH HELP

ZCHCH ?

Examples

The examples that follow provide the following information:

NR.
: is a 3-character record number.

F.A.
: is the file pool address.

I.D.
: is a record ID of 2 alphanumeric characters, a blank, and 4 hexadecimal characters.

The results of a forward chain chase are displayed in the following example.

```
User:   ZCHCH F4034035 F 002 HDR-4

System: CRZ10010I 21.46.00 CHAIN CHASE-FORWARD

        NR.       F.A.        I.D.

        1   00000000F4034035  A3 C1F3
        2   00000000F4034039  A3 C1F3
        CHAIN CHASE LIMIT REACHED

        2   RECORDS CHASED

        MORE RECORDS IN CHAIN
        END OF DISPLAY
```

The results of a backward chain chase are displayed in the following example.

```
User:   ZCHCH F4034035 B 002 HDR-4

System: CRZ10011I 21.46.00 CHAIN CHASE-BACKWARD

        NR.       F.A.        I.D.

        1   00000000F4034035  A3 C1F3
        2   00000000F4034031  A3 C1F3
        CHAIN CHASE LIMIT REACHED

        2   RECORDS CHASED

        MORE RECORDS IN CHAIN
        END OF DISPLAY
```

The results of a chain chase at the location specified for displacement are displayed in the following example. In this example, an 8-byte file address in 4x4 format and HDR-8 were specified; therefore, the 8-byte chain chase occurred.

```
User:   ZCHCH 00000000F4028801 U 002 064 HDR-8

System: CRZ10012I 11.51.49 CHAIN CHASE-DISPL.064

        NR.       F.A.        I.D.

        001 00000000F4028801  Y2 E8F2
        002 00000000000C0001  Y2 E8F2
        CHAIN CHASE LIMIT REACHED

        002 RECORDS CHASED

        NO MORE RECORDS IN CHAIN
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZCLAW ACTIVATE–Activate CLAW Workstations

Use this command to activate one or more Common Link Access to Workstation (CLAW) workstations.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW ACTivate── ──┬─WS-wsname──────────────────┬──────────────────────────►◄
                       │           ┌─────────────┐  │
                       │           ▼             │  │
                       ├─SDA-sda───┴──┬──────┬───┴──┤
                       │              └─.sda─┘      │
                       └─ALL────────────────────────┘
```

WS-*wsname*
: activates the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

SDA-*sda*
: activates the specified CLAW workstation, where *sda* is the 1- to 4-digit hexadecimal symbolic device address of the read channel unit on the CLAW workstation.

  Note: Symbolic device addresses are always assigned in read/write pairs. In the TPF system, the symbolic device address assigned to the read channel unit is always an ***even*** hexadecimal number (for example, X'20'). The write channel unit is always the following ***odd*** hexadecimal number (for example, X'21').

ALL
: activates all of the CLAW workstations that are defined for this TPF host processor.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCLAW HELP
  ZCLAW ?
- Although you can activate a CLAW workstation in 1052 state, socket applications are not activated until the TPF system is cycled to CRAS state or higher.
- If you perform an initial program load (IPL), the TPF system will automatically attempt to reactivate all of the active CLAW workstations.

Examples

The following information is displayed in the examples:

WS
: is the CLAW workstation name.

HOSTA
: is the CLAW host application name.

WSA
: is the CLAW workstation application name.

SDA
: is the symbolic device address of the read channel unit for the CLAW workstation.

The OS2TCP workstation is activated in the following example.

```
User:   ZCLAW ACT WS-OS2TCP

System: CLAW0037I 08.41.05 CLAW DEVICE ACTIVATION STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 ACTIVATED
        CLAW DEVICE ACTIVATION COMPLETED
```

The CLAW workstations that have symbolic device addresses 200 and 220 are activated in the following example.

```
User:   ZCLAW ACT SDA-200.220

System: CLAW0037I 08.41.05 CLAW DEVICE ACTIVATION STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 ACTIVATED
        WS-OS317202 HOSTA-TCPIP    WSA-API      SDA-0220 ACTIVATED
        CLAW DEVICE ACTIVATION COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about activating CLAW workstations and TCP/IP offload support.

ZCLAW ADD–Define CLAW Workstations and the CLAW Host Name

Use this command to do the following:
- Define a Common Link Access to Workstation (CLAW) workstation for this TPF host processor.
- Define the CLAW host name for this TPF host processor.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).
- Do not use this command if your loosely coupled complex has processors that are using a TPF image that includes 32-way loosely coupled processor support and other processors that are using an image that does not include 32-way loosely coupled processor support. Any CLAW workstations that are defined on a processor with 32-way loosely coupled processor support will be lost if that processor falls back to a previous level of support.

Format

```
►►──ZCLAW ADD── ──────────────────────────────────────────────────────────────►

►──┬─HOSTAppl-happlname── ──WSAppl-wapplname── ──WS-wsname── ──SDA-sda─┬────────►◄
   └─HOSTName-hostname──────────────────────────────────────────────────┘
```

HOSTAppl-*happlname*
is the 1- to 8-character alphanumeric name of the CLAW host application.

Note: If you are using a IBM 3172 Model 3 Interconnect Controller for your TCP/IP offload device, you must specify TCPIP for this parameter.

WSAppl-*wapplname*
is the 1- to 8-character alphanumeric name of the CLAW workstation application.

Note: If you are using a IBM 3172 Model 3 Interconnect Controller for your TCP/IP offload device, you must specify API for this parameter.

WS-*wsname*
is the 1- to 8-character alphanumeric name of the CLAW workstation.

Note: If you are using a IBM 3172 Model 3 Interconnect Controller for your TCP/IP offload device, you must specify OS2TCP or OS3172*xx* for this parameter, where *xx* is 2 unique alphanumeric characters.

SDA-*sda*
is the 1- to 4-digit hexadecimal symbolic device address of the read channel unit for the CLAW workstation.

Note: Symbolic device addresses are always assigned in read/write pairs. In the TPF system, the symbolic device address assigned to the read channel unit is always an ***even*** hexadecimal number (for example, X'20') and the write channel unit is always the following ***odd*** hexadecimal number (for example, X'21').

HOSTName-*hostname*
is the 1- to 8-character alphanumeric CLAW host name for this TPF host processor.

Note: If you are using a IBM 3172 Model 3 Interconnect Controller for your TCP/IP offload device, you must specify TCPIP for this parameter.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZCLAW HELP
ZCLAW ?

Examples

The CLAW host name is defined for this TPF host processor in the following example.

```
User:   ZCLAW ADD HOSTN-TCPIP

System: CLAW0044I 08.41.05 HOSTNAME SET AND SAVED IN CDT
```

A CLAW workstation is defined for this TPF host processor in the following example.

```
User:   ZCLAW ADD HOSTA-TCPIP WSA-API WS-OS2TCP SDA-200

System: CLAW0008I 08.41.05 ENTRY SUCCESSFULLY ADDED TO CLAW DEVICE TABLE
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about defining CLAW workstations, defining the CLAW host name, and TCP/IP offload support.

ZCLAW DELETE–Delete CLAW Workstation Definitions

Use this command to do the following:

- Delete a Common Link Access to Workstation (CLAW) workstation definition.
- Delete a definition for a specific CLAW host application and CLAW workstation application pair on a particular CLAW workstation.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).
- You cannot delete a definition for an active CLAW workstation. If the CLAW workstation is active, enter the ZCLAW INACTIVATE command to deactivate it before you enter this command.
- Do not use this command if your loosely coupled complex has processors that are using a TPF image that includes 32-way loosely coupled processor support and other processors that are using an image that does not include 32-way loosely coupled processor support. Any CLAW workstations that are deleted on a processor with 32-way loosely coupled processor support will have to be deleted again if that processor falls back to a previous level of support.

Format

```
►►──ZCLAW DELete── ──────────────────────────────────────────────────────────►

►──┬─WS-wsname─┬──────────────────────────────────────────────┬─┬──────────►◄
   │           └─ ──WSAppl-wapplname── ──HOSTAppl-happlname─┘ │
   └─SDA-sda──────────────────────────────────────────────────┘
```

WS-*wsname*
: deletes the definition for the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

WSAppl-*wapplname*
: deletes the definition for the specified CLAW workstation application, where *wapplname* is the 1- to 8-character alphanumeric name of the CLAW workstation application.

HOSTAppl-*happlname*
: deletes the definition for the specified CLAW host application, where *happlname* is the 1- to 8-character alphanumeric name of the CLAW host application.

SDA-*sda*
: deletes the definition for the CLAW workstation that has the specified symbolic device address, where *sda* is the 1- to 4-digit hexadecimal symbolic device address of the read channel unit on the CLAW workstation.

Note: Symbolic device addresses are always assigned in read/write pairs. In the TPF system, the symbolic device address assigned to the read channel unit is always an ***even*** hexadecimal number (for example, X'20') and the write channel unit is always the following ***odd*** hexadecimal number (for example, X'21').

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCLAW HELP
ZCLAW ?

Examples

The following information is displayed in the examples:

WS
is the CLAW workstation name.

HOSTA
is the CLAW host application name.

WSA
is the CLAW workstation application name.

SDA
is the symbolic device address of the read channel unit for the CLAW workstation.

The definition for the CLAW workstation application and CLAW host application pair (API and TCPIP) on the OS2TCP CLAW workstation is deleted in the following example.

```
User:   ZCLAW DELETE WS-OS2TCP WSA-API HOSTA-TCPIP

System: CLAW0026I 08.41.05 ZCLAW DEVICE DELETE STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 DELETED
        CLAW DEVICE DELETE COMPLETED
```

The definition for the OS317202 CLAW workstation is deleted in the following example.

```
User:   ZCLAW DELETE WS-OS317202

System: CLAW0026I 08.41.05 ZCLAW DEVICE DELETE STARTED
        WS-OS317202 HOSTA-TCPIP    WSA-API      SDA-0220 DELETED
        CLAW DEVICE DELETE COMPLETED
```

The definition for the CLAW workstation that has symbolic device address 240 is deleted in the following example.

```
User:   ZCLAW DELETE SDA-240

System: CLAW0026I 08.41.05 ZCLAW DEVICE DELETE STARTED
        WS-OS317203 HOSTA-TCPIP    WSA-API      SDA-0240 DELETED
        CLAW DEVICE DELETE COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about deleting CLAW workstation definitions and TCP/IP offload support.

ZCLAW DISPLAY–Display Information about CLAW Workstations

Use this command to display the following information:

- Definition for a specific CLAW workstation
- Definitions for all of the CLAW workstations defined on this TPF host processor
- Definitions for all of the active CLAW workstations defined for this TPF host processor
- CLAW host name defined for this TPF host processor.

Requirements and Restrictions

- You can enter this command only after Common Link Access to Workstation (CLAW) restart is completed.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW DISplay── ──┬─WS-wsname─┬──────────────────────────────────►◄
                      ├─ACTIVE────┤
                      ├─HOSTName──┤
                      └─ALL───────┘
```

WS-*wsname*
displays the definition for the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

ACTIVE
displays the definitions for all of the active CLAW workstations defined for this TPF host processor.

HOSTName
displays the CLAW host name that was defined for this TPF host processor.

ALL
displays the definitions for all of the CLAW workstations defined for this TPF host processor.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZCLAW HELP
ZCLAW ?

Examples

The following information is displayed in the examples:

WS
is the CLAW workstation name.

HOSTA
is the CLAW host application name.

WSA
is the CLAW workstation application name.

SDA
: is the symbolic device address of the read channel unit for the CLAW workstation.

ACTIVE
: indicates that the CLAW workstation is active. That is, the ZCLAW ACTIVATE command was entered for the CLAW workstation.

CONNECTED
: indicates that the CLAW workstation is active and 1 or more TPF socket applications are connected to the CLAW workstation application on that CLAW workstation.

The definition for the OS2TCP CLAW workstation is displayed in the following example.

```
User:   ZCLAW DISPLAY WS-OS2TCP

System: CLAW0010I 08.41.05 CLAW DEVICE DISPLAY STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 ACTIVE
        CLAW DEVICE DISPLAY COMPLETED
```

The definitions for the active CLAW workstations defined for this TPF host processor are displayed in the following example.

```
User:   ZCLAW DISPLAY ACTIVE

System: CLAW0010I 08.41.05 CLAW DEVICE DISPLAY STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 CONNECTED
        WS-OS317202 HOSTA-TCPIP    WSA-API      SDA-0220 ACTIVE
        CLAW DEVICE DISPLAY COMPLETED
```

The definitions for all of the CLAW workstations defined for this TPF host processor are displayed in the following example.

```
User:   ZCLAW DISPLAY ALL

System: CLAW0010I 08.41.05 CLAW DEVICE DISPLAY STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 CONNECTED
        WS-OS317202 HOSTA-TCPIP    WSA-API      SDA-0220 ACTIVE
        CLAW DEVICE DISPLAY COMPLETED
```

The CLAW host name that was defined for this TPF host processor is displayed in the following example.

```
User:   ZCLAW DISPLAY HOSTN

System: CLAW0046I 08.41.05 TPF HOSTNAME IS TCPIP
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about CLAW workstation definitions, the CLAW host name, and TCP/IP offload support.

ZCLAW INACTIVATE–Deactivate CLAW Workstations

Use this command to deactivate one or more CLAW workstations.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW INACTivate── ──┬─WS-wsname──┬──┬─────────┬──────────────────►◄
                         ├─SDA-sda────┤  └──FORCE──┘
                         └─ALL────────┘
```

WS-*wsname*
deactivates the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

SDA-*sda*
deactivates the CLAW workstation that has the specified symbolic device address, where *sda* is the 1- to 4-digit hexadecimal symbolic device address of the read channel unit on the CLAW workstation.

Note: Symbolic device addresses are always assigned in read/write pairs. In the TPF system, the symbolic device address assigned to the read channel unit is always an ***even*** hexadecimal number (for example, X'20') and the write channel unit is always the following ***odd*** hexadecimal number (for example, X'21').

ALL
deactivates all of the active CLAW workstations that are defined for this TPF host processor.

FORCE
forces the deactivation of the specified CLAW workstations without allowing the socket applications to run cleanup routines.

Notes:

1. Specify this parameter only if a previous request to deactivate a CLAW workstation failed.
2. Use this parameter only in a test environment because the results cannot be predicted.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCLAW HELP
  ZCLAW ?
- Use this command to deactivate a CLAW workstation before you delete it using the ZCLAW DELETE command.

- If the TPF system is cycled below CRAS state, the socket connections are automatically deactivated, regardless of whether the CLAW workstation is active or not.

Examples

The following information is displayed in the examples:

WS
: is the CLAW workstation name.

HOSTA
: is the CLAW host application name.

WSA
: is the CLAW workstation application name.

SDA
: is the symbolic device address of the read channel unit for the CLAW workstation.

The OS2TCP CLAW workstation is deactivated in the following example.

```
User:   ZCLAW INACT WS-OS2TCP

System: CLAW0013I 08.41.25 CLAW DEVICE INACT STARTED
        WS-OS2TCP   HOSTA-TCPIP    WSA-API      SDA-0200 INACTIVATED
        CLAW DEVICE INACTIVATION COMPLETED
```

The CLAW workstation with symbolic device address 220 is deactivated in the following example.

```
User:   ZCLAW INACT SDA-220

System: CLAW0013I 08.41.25 CLAW DEVICE INACT STARTED
        WS-OS317202 HOSTA-TCPIP    WSA-API      SDA-0220 INACTIVATED
        CLAW DEVICE INACTIVATION COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about deactivating a CLAW workstation and about TCP/IP offload support.

ZCLAW RESET–Reset the ZCLAW Command Lock

Use this command to reset the ZCLAW command lock.

Attention: Use the ZCLAW RESET command only in a test environment because the results cannot be predicted.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW RESET──────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCLAW HELP
  ZCLAW ?
- Each time you enter a ZCLAW command, the TPF system sets a lock that prevents you from entering another ZCLAW command until processing is completed for the previous ZCLAW command.

Examples

The ZCLAW command lock is reset in the following example.

```
User:   ZCLAW INACT WS-OS2TCP

System: CLAW0030E 08.30.12 ZCLAW PROCESSING STILL ACTIVE FOR A PREVIOUS MESSAGE

User:   ZCLAW RESET

System: CLAW0047I 08.41.25 ZCLAW RESET PROCESSING COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about the ZCLAW command lock and TCP/IP offload support.

ZCLAW STATUS–Display CLAW Status Information

Use this command to display the following status information about the active CLAW workstations defined for this TPF host processor:
- Number of bytes sent and received
- Number of messages sent and received.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW STATus── ──┬─ALL────────────────────┬─────────────────────►◄
                     └─WS-wsname──┬──────────┬┘
                                  └─ ──LL-link─┘
```

ALL
displays status information about all of the active CLAW workstations defined for this TPF host processor.

WS-*wsname*
displays status information for the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

LL-*link*
displays status information only for the specified logical link on the CLAW workstation, where *link* is the number of the logical link from 1–31.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCLAW HELP
  ZCLAW ?
- Each time you enter this command, the status information for the specified CLAW workstation or logical link is reset to 0.

  That is, the first time you enter the ZCLAW STATUS command for a CLAW workstation or logical link, the status information that is displayed indicates the number of bytes and messages that were sent and received since that CLAW workstation was activated. If you enter the ZCLAW STATUS command for that CLAW workstation or logical link again, the status information indicates the number of bytes and messages that were sent and received since the last time you entered the ZCLAW STATUS command.

Examples

Status information about the OS2TCP CLAW workstation is displayed in the following example. ADAPTER OPENED indicates that this status information contains the number of bytes and messages that were sent and received since the OS2TCP CLAW workstation was activated.

```
User:   ZCLAW STATUS WS-OS2TCP

System: CLAW0011I 08.41.25 CLAW DEVICE STATUS STARTED
        WS NAME      SENT- BYTES MESSAGES  RECEIVED- BYTES MESSAGES
        SINCE
        ------------------------------------------------------------
        OS2TCP            254670     2687           245873     2640
        ADAPTER OPENED

        CLAW DEVICE STATUS COMPLETED
```

Status information is displayed ***again*** for the OS2TCP CLAW workstation. Notice that the status information contains only the number of bytes and messages that was sent and received since the time that the ZCLAW STATUS command was entered in the previous example.

```
User:   ZCLAW STATUS WS-OS2TCP

System: CLAW0011I 08.43.29 CLAW DEVICE STATUS STARTED
        WS NAME      SENT- BYTES MESSAGES  RECEIVED- BYTES MESSAGES
        SINCE
        ------------------------------------------------------------
        OS2TCP            127334     1344           122936     1320
        FRI MAR 15 08.41.25 1996

        CLAW DEVICE STATUS COMPLETED
```

Status information about logical link 3 on the OS2TCP CLAW workstation is displayed in the following example.

```
User:   ZCLAW STATUS WS-OS2TCP LL-3

System: CLAW0011I 08.44.02 CLAW DEVICE STATUS STARTED
        WS NAME  LL     SENT- BYTES MESSAGES  RECEIVED- BYTES MESSAGES
        SINCE
        ---------------------------------------------------------------
        OS2TCP   03              0       0               0       0
        FRI MAR 15 08.43.29 1996

        CLAW DEVICE STATUS COMPLETED
```

Status information about all of the active CLAW workstations defined for this TPF host processor is displayed in the following example.

```
User:   ZCLAW STATUS ALL

System: CLAW0011I 08.49.36 CLAW DEVICE STATUS STARTED
        WS NAME      SENT- BYTES MESSAGES  RECEIVED- BYTES MESSAGES
        SINCE
        ------------------------------------------------------------
        OS2TCP            254670     2687           245873     2640
        FRI MAR 15 08.44.02 1996

        OS317202          325496     3115           314568     3068
        ADAPTER OPENED

        CLAW DEVICE STATUS COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP offload support.

ZCLAW TRACE–CLAW Trace Function

Use this command to do the following:

- Start and stop the Common Link Access to Workstation (CLAW) data trace or process trace function
- Start and stop writing the CLAW trace information to tape
- Display the status of the CLAW trace function.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZCLAW TRACE── ──┬─START── ──┬─ALL───────┬──┬──────────────────────────►◄
                    │           ├─WS-wsname─┤  │
                    │           ├─TAPE──────┤  │
                    │           └─PROCESS───┘  │
                    ├─STOP── ──┬─ALL─┬─────────┤
                    │          └─TAPE┘         │
                    └─STATUS───────────────────┘
```

START
: starts the CLAW data trace or process trace function or starts writing the CLAW trace information to tape.

ALL
: starts the CLAW data trace function for all of the CLAW workstations defined for this TPF host processor.

WS-*wsname*
: starts the CLAW data trace function for the specified CLAW workstation, where *wsname* is the 1- to 8-character alphanumeric name of the CLAW workstation.

TAPE
: starts writing the CLAW trace information to the real-time (RTA or RTL) tape.

PROCESS
: starts the CLAW process trace function for all of the CLAW workstations defined for this TPF host processor.

STOP
: stops the CLAW trace function or stops writing the CLAW trace information to tape.

ALL
: stops the CLAW trace function on all of the CLAW workstations where it was started and stops writing the CLAW trace information to tape.

TAPE
: stops writing the CLAW trace information to tape.

STATUS
displays the status of the CLAW trace function.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCLAW HELP
  ZCLAW ?
- Use the CLAW data trace postprocessor to format and print the CLAW data trace information that was written to tape.
- Use the CLAW process trace postprocessor to format and print the CLAW process trace information that was written to tape.

Examples

The CLAW data trace function is started for the OS2TCP CLAW workstation in the following example.

```
User:   ZCLAW TRACE START WS-OS2TCP

System: CLAW0089I 08.41.25 ZCLAW TRACE STARTED ON WORKSTATION OS2TCP
```

The CLAW data trace information is written to tape in the following example.

```
User:   ZCLAW TRACE START TAPE

System: CLAW0084I 08.41.25 ZCLAW TRACE STARTED ON TAPE
```

The status of the CLAW data trace function is displayed in the following example.

```
User:   ZCLAW TRACE STATUS

System: CLAW0097I 08.41.25 ZCLAW TRACE STATUS DISPLAY
        ZCLAW TRACE TAPE ACTIVE
        ZCLAW TRACE ACTIVE ON WORKSTATION OS2TCP
```

The CLAW data trace function is stopped and the CLAW data trace information is no longer written to tape in the following example.

```
User:   ZCLAW TRACE STOP ALL

System: CSMP0097I 08.41.25 CPU-B SS-BSS  SSU-HPN  IS-01
        CLAW0094I 08.41.25 ZCLAW TRACE STOPPED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about the CLAW trace function, CLAW trace postprocessors, and TCP/IP offload support.

ZCNIS–Change Instruction-Stream Parameters

Use this command to change the values used in a tightly coupled processor. You can change the operating mode for future initial program loads (IPLs) or you can set the values of the tuning parameters used by the I-stream scheduler.

Requirements and Restrictions

None.

Format

```
►►──ZCNIS── ──┬─MP─────────────────┬──────────────────────────────►◄
              ├─UP─────────────────┤
              ├─DISplay────────────┤
              └─┬─MAXq──┬─-value───┘
                └─REAct─┘
```

MP

updates keypoint I to generate a multiprocessing environment for the next IPL. This is the normal operating mode for the TPF system and allows the TPF system to use all the available I-streams.

Note: You cannot specify this parameter if the processor has only one I-stream or if only one I-stream can be active on the processor at a time. In addition, you cannot specify this parameter if the program test vehicle (PTV) is active because PTV is restricted to run only in uniprocessing mode.

UP

updates keypoint I to generate a uniprocessing environment for the next IPL. This forces the TPF system to run as though there was only one I-stream available. Use this mode only for trace functions, such as PTV, that do not operate correctly in MP mode.

DISplay

displays the current state of the environment.

MAXq

defines the maximum combined size of the CROSS, READY, and INPUT lists allowed before the I-stream scheduler overrides its normal algorithm. The value assigned to this parameter should be larger than any value reached during the normal steady state operation of the TPF system.

Note: Enter **ZSTAT U** to display the current size of the CROSS, READY, and INPUT lists.

REAct

specifies how rapidly the I-stream scheduler reacts to a change in work load composition; for example, starting a utility. Lower values cause the schedule to react faster.

Attention: If you change this value from the default value that is set by the TPF system, make the changes in small increments and observe the results carefully. If the value assigned to this parameter is too large, the TPF system makes corrections very slowly. If the value assigned to this parameter is too small, the TPF system may overreact to work load changes and prevent I-streams from balancing correctly.

value
is the hexadecimal value, from X'00'–X'7FFFFFFF', to assign to the MAXQ or REACT parameter.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCNIS HELP
  ZCNIS ?
- In a high performance option (HPO) loosely coupled complex, the ZCNIS command sets the operating mode and scheduler parameters for the current processor, but not for the other processors in the complex.

Examples

The current state of the environment is displayed in the following example.

```
User:   ZCNIS DISP

System: CNIS0004I 13.39.10 CURRENTLY RUNNING IN UP MODE
        CNIS0062I 13.39.10 I-STREAM SCHEDULER CONSTANTS: REACT-00001000, MAXQ-00000064
```

Related Information

See *TPF Concepts and Structures* and *TPF System Generation* for more information about I-streams, uniprocessors, and multiprocessors.

ZCNTM–Start Macro Counting for the Real-Time Trace Utility

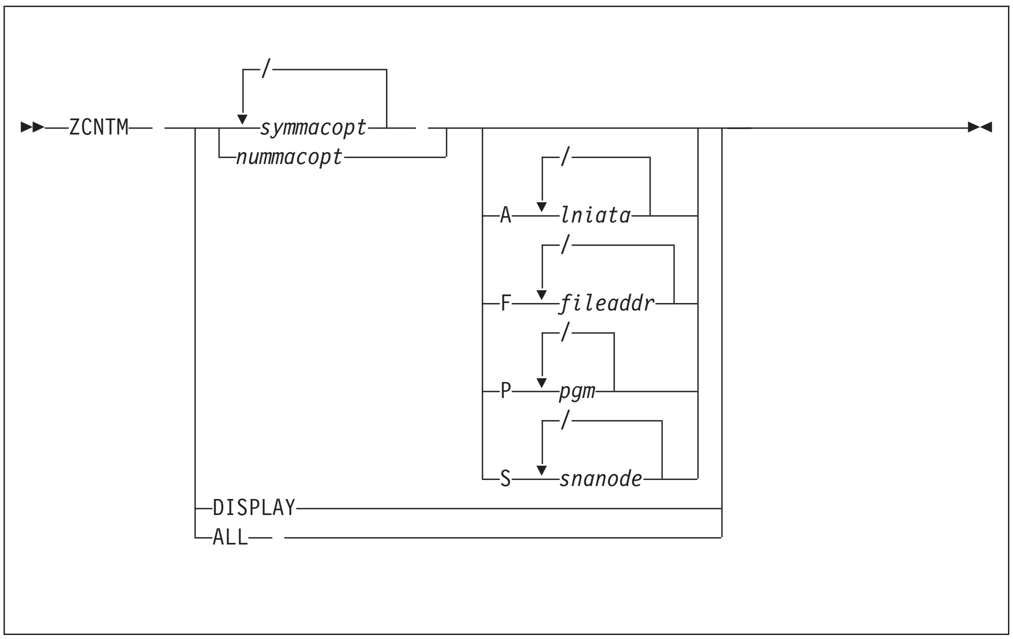
Use this command to start the macro counting function of the real-time trace (RTT) utility. The RTT utility counts or traces the use of specific macros and related system activity. The ZCNTM command creates a list of the sleected macros and provides a total of executions for each macro during the count period.

Use the ZTRAC command to start the macro tracing function. Use the ZSTOP command to stop the macro tracing or macro counting function.

Requirements and Restrictions

Each ZCNTM command must have a corresponding ZSTOP command. If you enter a second ZCNTM command before the count period is ended with a ZSTOP command, the second ZCNTM command is rejected.

Format



symmacopt
is the symbolic name for the macro trace option. Specify one or more of the following:

ENBK
counts enter-type macros and the BACKC macro.

CRET
counts the DEFRC and DLAYC, and create-type macros.

CTRL
counts the EXITC, LMONC, MONTC, and WAITC macros.

FILE
counts file-type macros and the UNFRC macros.

FIND
counts find-type macros.

DATA
counts the FLIPC, GETCC, and RELCC macros.

POOL
counts get file storage-type macros and the RELFC macro.

SEND
counts the CRASC macro and send-type macros.

RTAP
counts the real-time tape macros.

GTAP
counts the AMSSC and FDCTC macros, and the general tape macros.

MISC
counts miscellaneous macros.

SON
counts SON macros.

nummacopt
is the numeric value for the macro trace option. Specify one of the following:

0 counts SXSVC macros.

1 counts enter-type macros and the BACKC macro.

2 counts the DEFRC and DLAYC, and create-type macros.

3 counts the EXITC, LMONC, MONTC, and WAITC macros.

4 counts file-type macros and the UNFRC macros.

5 counts find-type macros.

6 counts the FLIPC, GETCC, and RELCC macros.

7 counts get file storage-type macros and the RELFC macro.

8 counts the CRASC macro and send-type macros.

9 counts the real-time tape macros.

A counts the AMSSC and FDCTC macros, and the general tape macros.

B counts miscellaneous macros.

D counts SON macros.

F counts all macros.

You ***cannot*** combine numeric macro options. If you want to specify more than one macro option for a command, use the symbolic macro option values.

A*lniata*
specifies the A trace mode. The A trace mode limits the RTT utility to activity assoiated with messages from specified terminals, where *lniata* is the line number, interchange address, and terminal address (LNIATA). If you specify this parameter, the RTT utility is activated whenever the specified macro is issued by a program activated as a result of a message from one of the identified terminals. Keep the following in mind when specifying the value for *lniata*:

- Specify 2, 4, or 6 hexadecimal characters. A 2-character value represents a line number for LNIATAs. A 4-character value represents a line/interchange for LNIATAs. A 6-character value represents the entire LNIATA.
- You can specify multiple LNIATAs on the same command (separated by the / character), but the LNIATAs must all have the same length. SLC link numbers may be specified as a line number, right-justified, and padded by zeros to meet the 4-character or 6-character length requirement.

- You can specify synchronous data link control (SDLC) pseudo LNIATAs in the same way as actual LNIATAs. You can specify locally attached 3270s in the same way as line numbers. The IA and TA are both X'00'. You can specify SDLC pseudo line numbers for SDLC SNA devices, which must be 6 characters long. The pseudo LNIATA is used to refer to an SDLC SNA device (network control program (NCP), line, cluster controller, or logical unit).
- References can be made to different devices on the same command.
- SDLC pseudo line numbers can to intermixed with LNIATAs on the same command. The SDLC pseudo line numbers must be 6 characters long. The LNIATAs can be 2, 4, or 6 characters long. Any SDLC pseudo line numbers specified are not counted in the maximum number of LNIATAs that can be specified.
- If you do not specify particular address parameters, you can use the ALL parameter to trace input from all terminal addresses (including SDLC pseudo lines).

F*fileaddr*
specifies the F trace mode. The F trace mode limits the RTT utility to activity associated with specified file addresses, where *fileaddr* is a 4- or 8-byte file address. If you specify this parameter, the RTT utility is activated whenever file-type macros use the identified file addresses. You can specify the ALL parameter to trace all ECBs that reference file addresses and these file addresses can be 4- or 8-byte file addresses. You can specify multiple file addresses (separated by the / character); however, the specified file addresses must have the same length: all 4- or 8-byte addresses.

P*pgm*
specifies the P trace mode. The P trace mode limits the RTT utility to activity of the macros issued by specified programs, where *pgm* is the 1- to 4-character name of the program to be traced or counted. If you specify this parameter, the RTT utility is activated whenever the specified macro is issued by the identified program. You can specify multiple program names (separated by the / character) on one command; however, all the names must be the same length: all 1, 2, 3, or 4 characters.

S*snanode*
specifies the S trace mode. The S trace mode limits the RTT utility to activity associated with messages from specified SNA node names, where *snanode* is the 1- to 8-character node name of the SNA logical unit (LU). Keep the following in mind when specifying the value for *snanode*:

- The node names must be associated with the CPU running the RTT utility.
- You can specify multiple SNA node names on the same command (separated by the / character), but length of the command, including the action code and end-of-message (EOM) character does not exceed the device limit.
- You can use the ALL parameter to trace all ECBs generated by SNA input or output.

DISPLAY
displays all symbolic parameters.

ALL
specifies all macro options. This is the same as entering **ZCNTM F**, using the numeric F for all macro options.

Note: You must leave a blank space after the ALL parameter for the command to process correctly.

Additional Information

All output from the RTT utility is written to the real-time tape (RTL/RTA). Use the offline diagnostic output formatter (DOF) utility to process the RTT output.

Examples

In the following example all macros are counted.

```
User:   ZCNTM ALL

System: RTT IN
```

In the following examples macros issued from programs beginning with the names ABC, EFC, and EFG are counted. The first example uses the symbolic macro parameter and the second example uses the numeric macro parameter. These are examples using the P trace mode.

```
User:   ZCNTM SEND PABC/EFC/EFG

System: RTT IN
```

```
User:   ZCNTM 8PABC/EFC/EFG

System: RTT IN
```

Related Information

- See “ZSTOP–Stop Macro Tracing or Macro Counting for the Real-Time Trace Utility” on page 1310 for more information about stopping the RTT utility.
- See “ZTRAC–Start Macro Tracing for the Real-Time Trace Utility” on page 1376 for more information about starting the macro tracing function.
- See *TPF Program Development Support Reference* for more information about the RTT utility and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline DOF utility.

ZCORO–Process a TPF Transaction Services Request

Use this command to display and control TPF transaction services processing.

Requirements and Restrictions

You can enter this command only in 1052 state or higher. TPF transaction services are not available until restart has processed the recovery log.

Format

```
►►──ZCORO── ──┬─Display─┬──────────────────────────────────────────────►◄
              ├─APPLY───┤
              └─NOAPPLY─┘
```

Display
displays the status of TPF transaction services processing.

APPLY
specifies that the restart program apply the recovery log.

NOAPPLY
specifies that the restart program not apply the recovery log. Use this parameter only during restart when it may be necessary to prompt you to decide whether to permit log manager (LM) restart to recover the log. There are two conditions under which this can happen:

- When CLM0 cannot reconnect to the CFLF of the prime module. The operator must determine if the recovery log can be applied without the CTKI keypoint being held. Holding CTKI prevents other processors from cycling through restart until this processor log can be recovered.
- When the CL21 program has detected that the subsystem (SS) or subsystem user (SSU) configuration has changed across the IPL. The operator must determine if applying the log will cause database corruption. The recovery log contains the subsystem ID associated with the record to be recovered. A change in the SS or SSU configuration can cause the SS or SSU IDs to be reassigned, resulting in records recovered to the wrong SS or SSU.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZCORO HELP
  ZCORO ?
- In the ZCORO command display, the value of TRACKS WRITTEN SINCE LAST IPL and BYTES WRITTEN SINCE LAST IPL is displayed in units of bytes; that is, either bytes, kilobytes, or gigabytes.
- The time stamp in the ZCORO command display is related to the high water mark of each field displayed.

Examples

The following example displays information about the status of TPF transaction services processing.

```
User:   ZCORO DISPLAY

System: CORO0001I 13.50.30 COMMIT SCOPE ACTIVITY

                  ALLOC  IN-USE  HIGHMARK  TIMESTAMP
                  -----  ------  ----------------------------------
        ECB IN C/S              6        13  Fri Sep  5 16:31:31 1997

        TRACK BUFFERS    10       1         3  Fri Sep  5 16:31:31 1997

        LOG TRACKS      420       1         5  Fri Sep  5 16:31:34 1997

        BSS LOG TRACKS  420       1         5  Fri Sep  5 16:31:34 1997


        TRACKS WRITTEN SINCE LAST IPL    52
        BYTES  WRITTEN SINCE LAST IPL       58.8 K


        COMMITS SINCE LAST IPL               45
        ROLLBACKS SINCE LAST IPL              1

        MAXIMUM BUFFERS ALLOWED IN A COMMIT SCOPE - NO LIMIT SET
        HIGH NUMBER RECORDED SINCE LAST IPL                   10+
```

The following example displays information about the status of TPF transaction services processing.

```
User:   ZCORO DISPLAY

System: CORO0001I 13.50.30 COMMIT SCOPE ACTIVITY

                  ALLOC  IN-USE  HIGHMARK  TIMESTAMP
                  -----  ------  ----------------------------------
        ECB IN C/S              0         8  Fri Sep  5 17:31:31 1997

        TRACK BUFFERS    10       0         2  Fri Sep  5 17:31:31 1997



        WP LOG TRACKS    50       1         5  Fri Sep  5 17:31:34 1997

        TRACKS WRITTEN SINCE LAST IPL        11.6 K
        BYTES  WRITTEN SINCE LAST IPL     38396.7 K


        COMMITS SINCE LAST IPL             7817
        ROLLBACKS SINCE LAST IPL              1

        MAXIMUM BUFFERS ALLOWED IN A COMMIT SCOPE           4000
        HIGH NUMBER RECORDED SINCE LAST IPL                   10+
```

Related Information

None.

ZCSON–Convert MCHR to FARF File Address

Use this command to convert an extended MCHR file address for a record to FARF3, FARF4, FARF5, and FARF6 format, depending on the system migration stage. Only the FARF address in the primary dispense mode is given.

Requirements and Restrictions

If files are duplicated, this command translates only the extended MCHR address of the prime copy.

Format

```
►►──ZCSON── ──mmmmccccchhhhrr── ──┬─F─┬──────────────────────────────────►◄
                                  └─P─┘
```

mmmmccccchhhhrr
: is a 14-digit extended MCHR file address, where:

mmmm
: is the 4-digit hexadecimal symbolic module number.

cccc
: is the 4-digit hexadecimal cylinder number.

hhhh
: is the 4-digit hexadecimal head number.

rr
: is the 2-digit hexadecimal record number.

F Specifies that the file address is for a fixed record.

P Specifies that the file address is for a pool record.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZCSON HELP

ZCSON ?

Examples

In the following example, the specified MCHR file address is converted to FARF addressing format.

```
User:   ZCSON 0047002C000004 P

System: CSON0002I 08.07.20 FARF ADDRESS IS 0000000000100000
```

Related Information

See *TPF Database Reference* for more information about file address formats.

ZCTKA ALTER–Alter Storage Allocation Values

Use this command to change the following values:

- Storage allocation values that were initially defined using the system initialization program (SIP) and then moved to keypoint A.
- Timeout value used to monitor stalled tape module queues.
- Heap storage management values.
- Thread management values.

Requirements and Restrictions

- You can enter this command only from the prime CRAS.
- If the multiple database function (MDBF) was generated, you can enter this command from any subsystem. However, you can change certain subsystem-shared values such as PHOLD, OHOLD, RLBUF, RLVAL, MCMTB, AVOIDT, TAPQTIME, MTHD, and TSTK only from the basic subsystem (BSS).
- Changes that are made using this command affect only the current processor. The storage allocation values on other processors remain unchanged.
- If more than one subsystem was generated, you cannot change the block allocation values for the entire TPF system using this command. You can, however, see the block allocation values by entering the ZSTAT command.
- Only system programmers should use this command.

Format

```
              ┌─────────────────────────────┐
              ▼       (1)                   │
►►──ZCTKA Alter──────────┬─AP24──────┬─-value─┴──┬──────────────────────────┬──►◄
                         ├─AP31──────┤           └──NEST─┬─NOLIMIT─┬────────┘
                         ├─AVOIDT────┤                   └─LIMIT───┘
                         ├─CMB───────┤
                         ├─CRETM─────┤
                         ├─CRETS─────┤
                         ├─ECB───────┤
                         ├─EMPS──────┤
                         ├─ESPS──────┤
                         ├─FRM───────┤
                         ├─IOB───────┤
                         ├─ISAI──────┤
                         ├─ISAS──────┤
                         ├─MCMTB─────┤
                         ├─MMHS──────┤
                         ├─MSHS──────┤
                         ├─MTHD──────┤
                         ├─OHOLD─────┤
                         ├─PHOLD─────┤
                         ├─RLBUF─────┤
                         ├─RLVAL─────┤
                         ├─SSPS──────┤
                         ├─SWB───────┤
                         ├─TAPQTIME──┤
                         ├─TSTK──────┤
                         └─XPAT──────┘
```

Notes:

1 You can also separate the parameters with a comma (,).

AP24
changes the size of the application program area that was allocated for core resident programs that run in 24-bit addressing mode.

AP31
changes the size of the application program area that was allocated for core resident programs that run in 31-bit addressing mode.

AVOIDT
changes the number of times an application will avoid an application timeout between loss of control. Specify a value of 0-32766.

Notes:

1. A value of 0 means the ECB will time out using the standard system timeout and will not avoid timeouts any additional times.
2. Specify the minimum value needed. A large value can cause system performance problems or lockout problems.
3. The value specified in the AVOIDT parameter acts as a multiplier of the application timeout value. For example, if the standard application timeout value is 500 milliseconds and the value specified in the AVOIDT parameter

is 5, an ECB performing C function trace will wait 5 times the standard timeout in addition to the 1 time for the standard timeout for a total of ((5 times 500) plus 500) or 3000 milliseconds.

4. Changing the AVOIDT value will have an effect on the timeout of ISO-C programs that have been compiled using the TEST option of one of the IBM C/370 family of compilers supported by the TPF 4.1 system, and that are traced using the C function trace facility.
5. The value specified in the AVOIDT parameter is placed in keypoint A. TPF system code can use the contents of the CINFC CMMAVMAX field for the contents of the SETOC macro AVOIDT parameter register when issuing the SETOC macro.

CMB
changes the number of common blocks that were allocated.

CRETM
changes the number of slots that were allocated in the CRET minutes table.

CRETS
changes the number of slots that were allocated in the CRET seconds table.

ECB
changes the number of ECBs that were allocated.

EMPS
changes the maximum size of the ECB heap. Specify the size in megabytes (MB).

ESPS
changes the maximum size of the ISO-C stack. Specify the size in megabytes (MB).

FRM
changes the number of frames that were allocated.

IOB
changes the number of I/O control blocks (IOBs) that were allocated.

ISAI
changes the number of 4-KB frames acquired during stack overflow processing for the ISO-C stack.

ISAS
changes the number of 4-KB frames acquired for initial ISO-C stack allocation.

MCMTB
changes the maximum number of commit buffers that are permitted in a single transaction. A value of 0 for this parameter indicates that there is no user-set maximum; therefore, there is no limit to the number of commit scope buffers in a transaction.

MMHS
changes the maximum number of 4-KB frames that an ECB can acquire for heap storage. The amount of storage that can be acquired cannot be more than the maximum size of the ECB heap, which is set by the EMPS parameter. For example, if the EMPS parameter is set to 1, this parameter cannot be more than 256, because 256 4-KB frames comprise 1MB of storage.

MSHS
changes the maximum number of 4-KB frames that an ECB can acquire for its ISO-C stack. The amount of storage that can be acquired cannot be more than the maximum size of the ISO-C stack, which is set by the ESPS parameter. For

example, if the ESPS parameter is 1, this parameter cannot be more than 256, because 256 4-KB frames comprise 1MB.

MTHD
changes the maximum number of threads allowed for a process.

OHOLD
changes the number of overflow hold table entries that were allocated.

PHOLD
changes the number of primary hold table entries that were allocated.

RLBUF
changes the number of buffers that are used for recovery log I/O operations. You can specify a value from 10–9999 for this parameter.

Note: If you enter the ZCTKA DISPLAY command and the value displayed for the RLBUF field is 0, that value indicates that you did not alter the number of the buffers and the TPF system is using the default value of 10.

RLVAL
changes the percentage value that is used to reduce the number of records allocated to the recovery log so that the recovery log will fit in the VPARS buffer area. You can specify a value from 10 to 99 for this parameter.

SSPS
changes the size of the system heap (in megabytes).

SWB
changes the number of system work blocks (SWBs) that were allocated for use by the online TPF system.

TAPQTIME
changes the number of seconds that may elapse before a stalled tape module queue condition is reported. A value of 0 indicates that stalled tape module queue monitoring will not occur. The maximum value for this parameter is 255 seconds.

TSTK
changes the maximum number of 4-KB ISO-C stack frames for a thread. You can specify a value between 4–1024 that is a power of 2 for this parameter.

XPAT
changes the number of extra program allocation table (PAT) slots that were allocated for the E-type loader.

value
is the new storage allocation value. This value must be a decimal number.

Notes:

1. To determine the correct values for storage block and CP table allocation parameters, see the information about processor and main storage resources in *TPF System Generation*. Many of the default values for the storage block and CP table keywords can be found in the CORREQ and RAM macros, and in the documentation for these macros in *TPF System Generation*.
2. To determine the correct values for the tape processing resources, see the information about tape support in *TPF System Generation*.

NEST
specifies whether or not the number of program nesting levels is limited.

NOLIMIT
allows an unlimited number of program nesting levels.

LIMIT
limits the number of program nesting levels to the number that can fit in the ECB.

Additional Information

- Changes made using this command take effect the next time an initial program load (IPL) is performed on the TPF system.
- Enter the ZCTKA DISPLAY command to display the current storage allocation values and the timeout value.

Examples

The following example changes the number of IOBs, ECBs, SWBs, common blocks, and frames that were allocated during system generation. It also allows an unlimited number of program nesting levels.

```
User:   ZCTKA ALTER IOB-300 ECB-250 SWB-200 CMB-300 FRM-300 NEST-NOLIMIT

System: CTKA0017I 16.43.21  STORAGE ALLOCATIONS
        KEYWORD      OLD VALUE      NEW VALUE
         IOB               400            300
         ECB               200            250
         SWB               250            200
         CMB               200            300
         FRM               200            300
         EMPS                4              4
         ESPS                4              4
         MMHS              256            256
         MSHS              256            256
         ISAS                3              3
         ISAI                1              1
         SSPS                4              4
         MTHD              100            100
         TSTK              512            512
         XPAT             1000           1000
         AP24           921600         921600
         AP31           921600         921600
         CRETS              80             80
         CRETM             160            160
         NEST               35            NOL
         PHOLD              50             50
         OHOLD             120            120
         RLBUF              10             10
         RLVAL              50             50
         MCMTB             100            100
         AVOIDT              0              0
         TAPQTIME           10             10
        AN IPL MUST BE DONE TO IMPLEMENT THE NEW STORAGE ALLOCATION
```

Related Information

- See *TPF System Generation* for more information about the fields in keypoint A (CTKA) and SIP.
- See *TPF System Macros* for more information about the SETOC macro.

ZCTKA DISPLAY–Display Storage Allocation Value Queue Timeout Value

Use this command to display the following values:

- Storage allocation values that were initially defined using the system initialization program (SIP) and then moved to keypoint A.
- Timeout value used to monitor stalled tape module queues.
- Heap storage management values.
- Thread management values.

Requirements and Restrictions

- You can enter this command only from the prime CRAS.
- If the multiple database function (MDBF) was generated, you can enter this command from any subsystem. However, you can display certain subsystem-shared values such as PHOLD, OHOLD, RLBUF, RLVAL, MCMTB, AVOIDT, TAPQTIME, MTHD, and TSTK only from the basic subsystem (BSS).
- If more than one subsystem was generated, you cannot display the block allocation values for the entire TPF system by entering this command. You can, however, enter the ZSTAT command to display these values.

Format

```
                      ┌─ ──All────┐
►►──ZCTKA Display─────┼───────────┼──────────────────────────────────────────►◄
                      ├─ ──Block──┤
                      ├─ ──Table──┤
                      ├─ ──TAPe───┤
                      ├─ ──HEAP───┤
                      └─ ──THread─┘
```

All
displays the current storage allocation values.

Block
displays the current storage block allocations.

Table
displays the current control program (CP) table allocations.

TAPe
displays the current stalled tape module queue timeout value.

HEAP
displays heap storage management values.

THread
displays thread management values.

Additional Information

- The storage block allocation values displayed are the number of blocks allocated in the current subsystem.

- Enter the ZCTKA ALTER command to change the storage allocation and timeout values.

Examples

The current storage allocation values are displayed in the following example.

```
User:   ZCTKA DISPLAY

System: CTKA0018I 11.09.00  STORAGE ALLOCATIONS

        KEYWORD       ALLOCATION
         IOB                500
         ECB                400
         SWB                250
         CMB                 75
         FRM                600
         EMPS                 4
         ESPS                 1
         MMHS               256
         MSHS               256
         ISAS                 3
         ISAI                 1
         SSPS                10
         MTHD               100
         TSTK               512
         XPAT              1000
         AP24           2000000
         AP31           3000000
         CRETS               80
         CRETM              160
         NEST               NOL
         PHOLD              257
         OHOLD             1001
         RLBUF               10
         RLVAL               50
         MCMTB              100
         AVOIDT               0
         TAPQTIME            10
        END OF DISPLAY
```

The current storage block allocations are displayed in the following example.

```
User:   ZCTKA DISPLAY BLOCK

System: CTKA0018I 15.28.54  STORAGE ALLOCATIONS

        KEYWORD       ALLOCATION
         IOB                500
         ECB                400
         SWB                250
         CMB                 75
         FRM                600
        END OF DISPLAY
```

The current heap management values are displayed in the following example.

```
User:   ZCTKA DISPLAY HEAP

System: CTKA0018I 16.27.49  STORAGE ALLOCATIONS

        KEYWORD       ALLOCATION
         EMPS                4
         ESPS                1
         MMHS              256
         MSHS              256
         ISAS                3
         ISAI                1
         SSPS               10
        END OF DISPLAY
```

The current thread management values are displayed in the following example.

```
User:   ZCTKA DISPLAY THREAD

System: CTKA0018I 15.44.19  STORAGE ALLOCATIONS

        KEYWORD       ALLOCATION
         MTHD              100
         TSTK              512
        END OF DISPLAY
```

Related Information

See *TPF System Generation* for more information about the fields in keypoint A (CTKA) and SIP.

ZCYCL–Cycle Down or Cycle Up (Change the System Operating State)

Use this command to change from one system state to another.

Requirements and Restrictions

None.

Format

```
►►──ZCYCL── ──┬─1052─┬──┬──┬───────┬──┬───────┬──┬──────────────────────────►◄
              ├─CRAS─┤  │  └─ALL───┘  └─BP────┘  │
              ├─MESW─┤  │                ┌─/─────┐ │
              ├─NORM─┤  │                ▼       │ │
              └─UTIL─┘  └──┬─INC─┬── ────ssname──┴─┘
                           └─EXC─┘
```

1052
cycles the TPF system to 1052 state.

CRAS
cycles the TPF system to CRAS state.

MESW
cycles the TPF system to MESW state.

NORM
cycles the TPF system to normal (NORM) state.

UTIL
cycles the TPF system to utility (UTIL) state.

ALL
cycles all subsystems in a multiple database function (MDBF) environment. (You can also specify this parameter for a non-MDBF environment.)

BP
bypasses the check of active utility functions.

INC
cycles the specified subsystems.

Note: You can specify this parameter only in an MDBF environment.

EXC
cycles all the subsystems except the specified subsystems.

Note: You can specify this parameter only in an MDBF environment.

ssname
is the 1- to 4-character alphanumeric name of a subsystem (MDBF only). You can specify as many as 10 subsystem names.

Additional Information

- You can cycle to the next higher state when the CRAS is inactive by causing an external interrupt. However, when catastrophic error recovery is being performed, causing an external interrupt will only prevent the recovery.

- Enter the ZDSYS command to display the current system state.

Examples

The TPF system is cycled to NORM state in the following example.

```
User:   ZCYCL NORM

System: CYCL0001I 13.32.29 CYCL TO NORM - STARTED
        VFAC0015I 13.32.52 DELAYED FILES ENABLED

        CVCX0001I 13.32.52 SS BSS  NOW IN NORM STATE
```

Related Information

See *TPF Main Supervisor Reference* for more information about changing system states.

ZDADD–Display FARF File Addresses from Record Type and Ordinal Number

Use this command to display the file address of a record type and ordinal number. The file address is displayed in the following formats:
- FARF address
- MCHR address of the primary copy
- MCHR address of the duplicate copy, if it exists.

Requirements and Restrictions

None.

Format

```
►►──ZDADD── ──rectype.ordinal──┬─────────────┬─┬──────────────┬──────────────────►
                               └─ ──SSU-user─┘ └─ ──Cpuid-id─┘

►─┬──────────────────┬──────────────────────────────────────────────────────────►◄
  └─ ──Istream-xx───┘
```

rectype
   is a 1- to 4-digit hexadecimal record type.

ordinal
   is a 1- to 16-digit hexadecimal ordinal number.

SSU-*user*
   displays the file address for the specified subsystem user, where *user* is the 1- to 4-character alphanumeric subsystem user name.

CPUid-*id*
   displays the file address for the specified processor, where *id* is the 1-character alphanumeric CPU ID.

Istream-*xx*
   displays the file address for the specified I-stream, where *xx* is a decimal number from 1 to 16.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDADD HELP
  ZDADD ?
- The FARF address is displayed in the primary dispense mode if it is defined. Otherwise, the FARF address is displayed in the alternate dispense mode.

Examples

In the following example, the file address for ordinal 2 of record type 89 is displayed.

```
User:   ZDADD 89.2

System: DADD0004I 08.11.10 FARF ADDRESS-00000000F400B009
                           PRIME ADDRESS-0047000E000765
                           DUPLICATE ADDRESS-0048000E000765
```

Related Information

See *TPF System Generation* and *TPF Database Reference* for more information about record types, ordinals, and addressing formats.

ZDBRI RESTART–Restart Database Reorganization Input Phase

Use this command to restart the database reorganization (DBR) input phase at a specified record type. The tape is scanned until the specified record type is found. All remaining records on the tapes are loaded.

Requirements and Restrictions

- The TPF system must be in 1052 state.
- Before entering this command for fixed file records, perform an initial program load (IPL) for the general file.
- Before entering this command for pool records, perform an initial program load (IPL) for the prime module.

Format

```
►►──ZDBRI RESTART── ──┬─FIXED─┬── ──RECtype-recvalue─────────────────────►◄
                      └─POOL──┘
```

FIXED
specifies that the record is a fixed file record.

POOL
specifies that the record is a pool file record.

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Additional Information

If the record type is not found, DBR starts again at the first record type on the tape after the specified record type.

Examples

The input phase of DBR is restarted for all small duplicated pool records in the following example.

```
User:   ZDBRI RESTART POOL REC-SDP

System: DBRI0020I 19.45.07 DBR INPUT PHASE STARTED
```

The input phase of DBR is restarted for fixed file records beginning at record type 00D7 in the following example.

```
User:   ZDBRI RESTART FIXED REC-00D7

System: DBRI0020I 09.30.16 DBR INPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRI START–Start Database Reorganization Input Phase

Use this command to start the database reorganization (DBR) input phase for either fixed file or pool records.

Requirements and Restrictions

- The TPF system must be in 1052 state.
- Before entering this command for fixed file records, you must perform an initial program load (IPL) for the general file.
- Before entering this command for pool records, you must perform an initial program load (IPL) for the prime module.
- Load the logging tapes after the pool input is completed, if necessary.
- If there is more than one set of input tapes for either pool or fixed file records, run the input phase for each set of tapes. (DBR ends if an end-of-file condition occurs.)

Format

```
►►──ZDBRI START── ──┬─FIXED─┬──────────────────────────────────────────►◄
                    └─POOL──┘
```

FIXED
starts DBR for fixed file records.

POOL
starts DBR for pool records.

Additional Information

None.

Examples

The input phase for DBR is started for fixed file records in the following example.

```
User:   ZDBRI START FIXED

System: DBRI0020I 14.34.22 DBR INPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO ABORT–Stop Database Reorganization

Use this command to stop a database reorganization (DBR) run. If the tape is open, it is closed and all the control bits are reset.

Requirements and Restrictions

- Use this command only for the output phase of DBR.
- You can enter this command only when DBR is running.

Format

```
►►──ZDBRO ABORT──────────────────────────────────────────────────────────►◄
```

Additional Information

Use the ZDBRO RESTART command to restart DBR at a later time.

Examples

DBR is stopped in the following example.

```
User:   ZDBRO ABORT

System: DBRO0000I 19.45.28 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO BYPASS–Bypass Database Reorganization Record Types

Use this command to bypass a certain record type or to specify that specific ordinal number ranges in a record type are to be bypassed during database reorganization (DBR).

Requirements and Restrictions

- You can enter this command only before you start DBR.
- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►──ZDBRO BYPASS── ──┬─RECtype-recvalue──┬───────────────────────────┬─┬──────────►◄
                     │                   └─ ──RANge-startord.endord──┘ │
                     └─DIR─────────────────────────────────────────────┘
```

RECtype-*recvalue*
: specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
: Small, long-term

SST
: Small, short-term

SDP
: Small, long-term duplicate

LLT
: Large, long-term

LST
: Large, short-term

LDP
: Large, long-term duplicate

4LT
: 4-KB, long-term

4ST
: 4-KB, short-term

4DP
: 4-KB, long-term duplicate

4D6
: 4-KB, long-term FARF6 duplicate.

RANge-*startord***.***endord*
: bypasses an ordinal number range in the specified record type, where *startord* and *endord* are 1- to 16-digit hexadecimal numbers.

DIR
bypasses the pool records that are not in use.

Additional Information

If you specify the RECTYPE parameter without the RANGE parameter, all ordinal numbers in the specified record type are bypassed.

Examples

Ordinal numbers 110–119 are bypassed in the following example.

```
User:   ZDBRO BYPASS REC-0000 RAN-6E.77

System: DBRO0000I 12.54.00 REQUEST COMPLETE
```

All small, long-term records are bypassed in the following example.

```
User:   ZDBRO BYPASS REC-SLT

System: DBRO0000I 12.55.22 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO DISPLAY–Display Database Reorganization Record Types

Use this command to:
- Display the record types to be captured
- Display the record types that contain exceptions
- Display the exceptions for a specified record type
- Display the record types specified in the override record.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- Enter this command only before you start database reorganization (DBR). The results of the display are not valid after DBR is started.

Format

```
►►──ZDBRO DISPLAY── ──┬─┬─ALL─┬─┬────────────────────┬─┬──────────────►◄
                      │ ├─EXC─┤ └──RECtype-recvalue──┘ │
                      │ └─OVR─┘                        │
                      └─RECtype-recvalue─┬────────────────┬┘
                                         └──Start-ordnum──┘
```

ALL
displays the record types to be captured.

EXC
displays the record types containing exceptions.

OVR
displays the record types specified in the override record.

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Start-*ordnum*
specifies the starting set of exception ordinal numbers to be displayed, where *ordnum* is a number from 1 to FF.

Additional Information

- As many as 8 record types are displayed per line.
- In 1052 state, the display is truncated with `..T` after 44 record types are displayed because of the lack of pool records in that state.
- In NORM state, a maximum of 48 record types can be displayed. If there are more than 48 record types to be displayed, MORE is displayed on the last line.
- In either 1052 state or NORM state, to see the remaining record types, enter:

  ZDBRO DISPLAY ALL RECTYPE-*recvalue*

  where *recvalue* is the last record type that was displayed.

Examples

The record types to be captured are displayed in NORM state in the following example. Notice that only the first 48 record types in use are displayed. MORE, which is on the last line of the display, shows that there are more record types in use.

```
User:   ZDBRO DISPLAY ALL

System: DBRO0067I 10.22.21 RECORD TYPES TO BE CAPTURED
        0000 0001 0002 0003 0004 0005 0006 0007
        0008 0009 000A 000B 000C 000D 000E 000F
        0010 0011 0012 0013 0014 0015 0016 0017
        0018 0019 001A 001B 001C 001D 001E 001F
        0020 0021 0022 0023 0024 0025 0026 0027
        0028 0029 002A 002B 002C 002D 002E 002F
        MORE
```

The record types to be captured are displayed in 1052 state in the following example. Notice that only the first 44 record types in use are displayed. The ..T on the last line of the display shows that there are more record types in use.

```
User:   ZDBRO DISPLAY ALL

System: DBRO0067I 19.45.28 RECORD TYPES TO BE CAPTURED
        0005 0007 0012 0013 0014 0015 0016 0017
        0018 001A 001C 0023 0024 0025 0027 0028
        002A 002D 002E 002F 0030 0031 0032 0033
        0034 0035 0036 003A 003B 0040 0042 0043
        0044 0046 0047 0052 0059 005B 005D 0063
        0071 0073 0074 0076 0077..T
```

The record types that contain exceptions are displayed in the following example.

```
User:   ZDBRO DISPLAY EXC

System: DBRO0068I 19.45.28 RECORD TYPES WITH EXCEPTIONS
        0005   SST
```

The record types that are specified in the override record are displayed in the following example.

```
User:   ZDBRO DISPLAY OVR

System: DBRO0069I 19.45.28 RECORD TYPES IN OVERRIDE REC
        0005 0007
```

Record type 0005 is displayed in the following example.

```
User:   ZDBRO DISPLAY REC-0005

System: DBRO0077I 19.45.28 EXCEPTIONS FOR RECORD TYPE 0005
         START              END
        000000000000006E  0000000000000077
        0000000000000082  0000000000000083
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO IECB–Set Database Reorganization Input Phase ECB Count

Use this command to set the available entry control block (ECB) count for the input phase of database reorganization (DBR).

If the working keypoint was not initialized, a 4-KB core block is formatted and updated with the new ECB count and filed as the working keypoint. If the working keypoint was initialized, it is updated and refiled. If the input phase is running, the in-core keypoint is updated.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- You cannot enter this command if DBR is running for the input phase of fixed file records.
- Use this command carefully. ECBs are associated with frames. Increasing the number of ECBs increases the number of frames needed for DBR. See the *TPF Migration Guide: Program Update Tapes* for additional information about allocating frames for DBR.

Format

```
►►──ZDBRO IECB── ──ecbnum──────────────────────────────────────────────►◄
```

ecbnum
is the new available ECB count, from 01–99, to use during the input phase.

Additional Information

None.

Examples

The available ECB count is set to 20 in the following example.

```
User:   ZDBRO IECB 20

System: DBRO0000I 18.21.43 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO INIT–Initialize Database Reorganization Control Records

Use this command to initialize the control records that are used for database reorganization (DBR).

The DBR control records can be initialized to allow the capture of all fixed file and pool records in the system. Alternatively, the DBR control records can be initialized so that specific subsystem user, processor, and I-stream combinations are captured.

Requirements and Restrictions

- You must enter this command before you can access the DBR control records.
- If a previous DBR run has not ended, you cannot enter this command without the BYPASS parameter. For example, if you stop a DBR run (by using the ZDBRO ABORT command) before it is completed, you must do one of the following:
  - Restart the run and let it end before you enter this command
  - Enter this command and specify YES for the BYPASS parameter.

Format

```
                                                   ┌─BYPASS=NO──┐
►►─ZDBRO INIT─┬─────────────────────────┬─┼────────────┼──────────►◄
              └─ssu/processor/istream─┘   └─BYPASS=YES─┘
```

ssu/processor/istream
: is a triplet that specifies the subsystem user, processor, and I-stream combination. The specified I-stream is a decimal number from 1 to 16. This parameter captures all the records the specified subsystem user owns for that particular combination.

BYPASS
: specifies whether or not to override the control bit that is set to prevent the reinitialization of control records. Specify YES to reinitialize the control records before running DBR.

Attention: Use the BYPASS parameter with care because it reinitializes all DBR records regardless of contents. Using this parameter can cause loss of data that was previously entered by the installation. Any previous values that were set with other DBR commands are overwritten.

Additional Information

After initialization, you can use other commands associated with DBR to specify what sections of the database should not be captured during a DBR run.

Examples

In the following example, the control record is set to capture all the records owned by the SSU1 subsystem user on processor B and I-stream 2.

```
User:   ZDBRO INIT SSU1/B/2

System: DBRO0000I 13.42.33 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO OECB–Set Database Reorganization Output Phase ECB Count

Use this command to set the available entry control block (ECB) count for the database reorganization (DBR) output phase. If the output phase is not running, the master keypoint is updated with the new available ECB count. If the output phase is running, the core copy of the working keypoint is updated to reflect the new available ECB count.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- Use this command carefully. ECBs are associated with frames. Increasing the number of ECBs increases the number of frames needed for DBR. See the *TPF Migration Guide: Program Update Tapes* for additional information about allocating frames for DBR.

Format

```
►►──ZDBRO OECB── ──ecbnum──────────────────────────────────────────►◄
```

ecbnum
is the new available ECB count, from 01–99, to use during the output phase.

Additional Information

None.

Examples

The ECB count is set to 30 in the following example.

```
User:   ZDBRO OECB 30

System: DBRO0000I 14.33.21 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO RESET–Reset Database Reorganization Capture Settings

Use this command to reset the:

- Capture bit for a specific record type
- Exception ordinal number range for a specific record type
- In-use pool control bit
- DBR override record.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►—ZDBRO RESET—— ——┬RECtype-recvalue——┬──────────────────────────┬─┬──────────►◄
                   │                  └ ——RANge-startord.endord—┘ │
                   ├DIR─────────────────────────────────────────────┤
                   └OVR─────────────────────────────────────────────┘
```

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

RANge-*startord***.***endord*
removes the specified ordinal number range from the specified record type, where *startord* and *endord* are 1- to 16-digit hexadecimal numbers.

DIR
resets the bypass not-in-use bit to allow all pool records to be captured.

OVR
sets all entries in the DBR override keypoint to 0.

Additional Information

If you specify the RECTYPE parameter without the RANGE parameter, the capture bit for all ordinal numbers in the record type is set.

Examples

Records 22–31 in record type 0009 are reset to be captured in the following example.

```
User:   ZDBRO RESET REC-0009 RANGE-16.1F

System: DBRO0000I 07.30.32 REQUEST COMPLETE
```

All pool records are reset to be captured in the following example.

```
User:   ZDBRO RESET DIR

System: DBRO0000I 07.30.32 REQUEST COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO RESTART–Restart Database Reorganization Output Phase

Use this command to restart the database reorganization (DBR) output phase. Save the tape from the previous run for the input phase and mount a new tape when restart requests a tape.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- The TPF system must be in 1052 state unless record logging for the capture and restore utility is active. If logging is active, the system can be in NORM state when you enter this command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►──ZDBRO RESTART──┬────────────────────┬──────────────────────────►◄
                   └──RECtype-recvalue──┘
```

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Additional Information

- If you do not specify the RECTYPE parameter, the working keypoint is used to restart the DBR output phase with ordinal number 0 of the record type after the last record type that was successfully captured to tape.
- If you specify the RECTYPE parameter, the DBR output phase will restart at the first valid record type following the one specified. The record type specified must be the same type (fixed or pool) that was being processed when DBR was aborted.

Examples

DBR is restarted in the following example.

```
User:   ZDBRO RESTART

System: DBRO0012I 19.45.28 DBR OUTPUT PHASE RESTARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

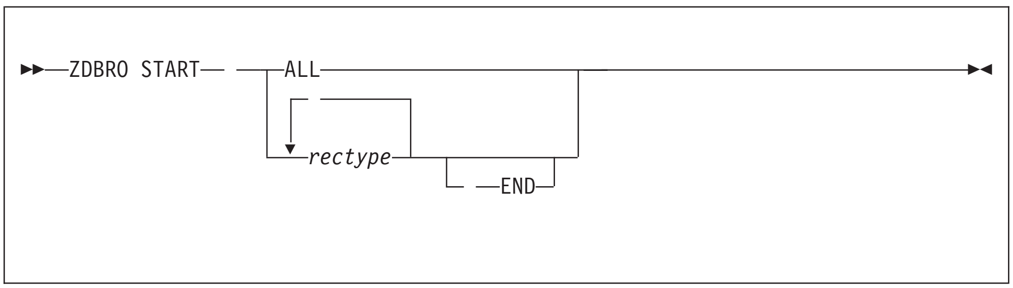
ZDBRO START–Start Database Reorganization Output Phase

Use this command to start the database reorganization (DBR) output phase. You can capture the database based on the settings in the master keypoint or set up the override keypoint to capture only part of the database. The partial capture starts when you enter this command with the END parameter.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- The TPF system must be in 1052 state unless record logging for the capture and restore utility is active. If logging is active, the system can be in NORM state when you enter this command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format



ALL
starts the database capture as specified by the master keypoint.

rectype
is a single record type or group of record types to be placed in the DBR override keypoint. The record type must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

You can specify as many as 8 record types. If more than 8 record types are required, enter the command with 8 record types and do not specify the END parameter; then enter the command again with the remaining record types followed by the END parameter. Processing does not begin until you specify the END parameter. If any previous ZDBRO START commands with the *rectype* parameter specified were entered, these records are also captured.

END
specifies that all record types to be placed in the override keypoint have been specified and starts the partial capture.

Additional Information

If you want to run the DBR output phase with buffered tape devices, you must do one of the following:

- Specify the DBRBUF=YES parameter on the CONFIG macro in your SIP stage 1 deck. See *TPF System Generation* for more information about the CONFIG macro.
- Enter the ZSYSG command with the DBRBUF parameter.

Attention: Running the DBR output phase with buffered tape devices will improve performance; however, if the system re-IPLs or tape errors occur while DBR is in progress, you may lose data. Also, because the switch that is set by the DBRBUF parameter is subsystem-shared, you can specify ZSYSG with the DBRBUF parameter only from the basic (BSS) subsystem.

Examples

In the following example, DBR is started for 4-KB long-term pool records.

```
User:   ZDBRO START 4LT END

System: DBRO0010I 07.30.32 DBR OUTPUT PHASE STARTED
```

In the following example, DBR is started for more than 8 fixed file types.

```
User:   ZDBRO START 0005 0007 0012 0013 0014 0015 0018 0019

System: DBRO0000I 15.40.05 REQUEST COMPLETE

User:   ZDBRO START 001A END

System: DBRO0010I 15.40.05 DBR OUTPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO STATUS–Display Database Reorganization Status

Use this command to display the following status information about the DBR run:

- Current record type
- Ordinal number
- Available entry control block (ECB) counts that are being processed by the database reorganization (DBR) output phase.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- DBR must be running when you enter this command.

Format

```
►►──ZDBRO STATUS──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

DBR status is displayed in the following example.

```
User:   ZDBRO STATUS

System: DBRO0062I 22.12.33 DBR IS PRESENTLY PROCESSING
        RECORD TYPE 0000,ORDINAL NUMBER 000000000000013D,ECB= 00000020
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBRO SWITCH–Restart Database Reorganization after Tape Error

Use this command to restart the database reorganization (DBR) run if an irrecoverable tape error occurs. If an irrecoverable tape error occurs, the current tape is closed, a message is sent to the console requesting a new tape, and DBR enters a defer loop. When the new tape is mounted, enter this command to restart DBR activity.

Requirements and Restrictions

- You must enter the ZDBRO INIT command before you enter any other ZDBRO command.
- You can enter this command only when you are prompted by the TPF system.
- Save the current tape for the input phase of DBR.

Format

```
►►──ZDBRO SWITCH──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

In the following example, DBR is restarted after a tape error occurred.

```
System: DBRO0015I 19.45.28 DBR HAS FINISHED PROCESSING RECORD TYPE 0145
        DBRO0085A 19.45.28 TAPE ERROR..SWITCH DBF, THEN ENTER ZDBRO SWITCH
        COTC0080A 19.45.28 TCLS HPN    REMOVE DBF FROM DEVICE 424
                           VSN A00243 G0004 S0001 D38K   SL  NOBLK  NOCOMP

User:   ZTMNT DBF 426 AO BP

System: COTM0008W 19.45.28 TMNT HPN    DEVICE 426 VSN A00245
                           UNEXPIRED FILE OVERWRITTEN
        COTM0046I 19.45.28 TMNT HPN    TAPE DBF MOUNTED ON DEVICE 426
                           VSN A00245 G0005 S0001 D38K   SL  NOBLK  NOCOMP

User:   ZDBRO SWITCH

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBRO0015I 19.45.28 DBR HAS FINISHED PROCESSING RECORD TYPE 0146
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSI RESTART–Restart Database Reorganization Input Phase

Use this command to restart the database reorganization (DBR) input phase at a specified record type. The tape is scanned until the specified record type is found. All remaining records on the tapes are loaded.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- The TPF system must be in 1052 state.
- Before entering this command for fixed file records, perform an initial program load (IPL) for the general file.
- Before entering this command for pool records, perform an initial program load (IPL) for the prime module.

Format

```
►►──ZDBSI RESTART── ──┬─FIXED─┬── ──RECtype-recvalue──────────────────────────►◄
                      └─POOL──┘
```

FIXED
specifies that the record is a fixed file record.

POOL
specifies that the record is a pool file record.

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Additional Information

None.

Examples

The input phase of DBR is restarted for all large short-term pool records in the following example.

```
User:   ZDBSI RESTART POOL REC-LST

System: DBRI0020I 19.45.07 DBR INPUT PHASE STARTED
```

In the following example, the input phase of DBR is restarted for fixed file records beginning at record type 3000.

```
User:   ZDBSI RESTART FIXED RECTYPE-3000

System: DBRI0020I 09.30.16 DBR INPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSI START–Start Database Reorganization Input Phase

Use this command to start the database reorganization (DBR) input phase for either fixed file or pool records.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- The TPF system must be in 1052 state.
- Before entering this command for fixed file records, perform an initial program load (IPL) for the general file.
- Before entering this command for pool records, perform an initial program load (IPL) for the prime module.
- Load the logging tapes after the pool input is completed, if necessary.
- If there is more than one set of input tapes for either pool or fixed file records, run the input phase for each set of tapes. (DBR ends if an end-of-file condition occurs.)

Format

```
►►──ZDBSI START── ──┬─FIXED─┬──────────────────────────────────────────►◄
                    └─POOL──┘
```

FIXED
starts DBR for fixed file records.

POOL
starts DBR for pool records.

Additional Information

None.

Examples

The input phase for DBR is started for fixed file records in the following example.

```
User:   ZDBSI START FIXED

System: DBSI0020I 14.34.22 DBR INPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO ABORT–Stop Database Reorganization

Use this command to stop a database reorganization (DBR) run. If the tape is open, it is closed and all the control bits are reset.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- Use this command only for the output phase of DBR.
- You can enter this command only when DBR is running.

Format

```
►►──ZDBSO ABORT──────────────────────────────────────────────────►◄
```

Additional Information

Use the ZDBSO RESTART command to restart DBR at a later time.

Examples

DBR is stopped in the following example.

```
User:   ZDBSO ABORT

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO BYPASS–Bypass Database Reorganization Record Types

Use this command to bypass a certain record type or to specify that specific ordinal number ranges in a record type are to be bypassed during database reorganization (DBR).

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You can enter this command only before you start DBR.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
>>──ZDBSO BYPASS── ──┬─RECtype-recvalue──┬──────────────────────────┬─┬──────><
                     │                   └─ ──RANge-startord.endord─┘ │
                     └─DIR────────────────────────────────────────────┘
```

RECtype-*recvalue*
: specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
: Small, long-term

SST
: Small, short-term

SDP
: Small, long-term duplicate

LLT
: Large, long-term

LST
: Large, short-term

LDP
: Large, long-term duplicate

4LT
: 4-KB, long-term

4ST
: 4-KB, short-term

4DP
: 4-KB, long-term duplicate

4D6
: 4-KB, long-term FARF6 duplicate.

RANge-*startord***.***endord*
bypasses an ordinal number range in the specified record type, where *startord* and *endord* are 1- to 16-digit hexadecimal numbers.

DIR
bypasses the pool records that are not in use.

Additional Information

If you specify the RECTYPE parameter without the RANGE parameter, all ordinal numbers in the specified record type are bypassed.

Examples

Ordinal numbers 120–123 in record type 0005 are bypassed in the following example.

```
User:   ZDBSO BYPASS REC-0005 RANGE-78.7B

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

All 4-KB short-term records are bypassed in the following example.

```
User:   ZDBSO BYPASS REC-4ST

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Ordinal numbers 2–5 in record type 0027 in a subsystem with multiple subsystem users are bypassed in the following example.

```
User:   WP/ZDBSO BYPASS REC-0027 RANGE-2.5

System: CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP2  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP3  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBSO0001I 21.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO DISPLAY–Display Database Reorganization Record Types

Use this command to do the following:
- Display the record types to be captured
- Display the record types that contain exceptions
- Display the exceptions for a specified record type
- Display the record types specified in the override record.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- Enter this command only before you start database reorganization (DBR). The results of the display are not valid after DBR is started.

Format

```
►►──ZDBSO DISPLAY── ──┬─┬─ALL─┬──┬───────────────────────┬──┬──────────────►◄
                      │ ├─EXC─┤  └──RECtype-recvalue──┘  │
                      │ └─OVR─┘                          │
                      └─RECtype-recvalue─┬───────────────┬┘
                                         └──Start-ordnum─┘
```

ALL
displays the record types to be captured.

EXC
displays the record types containing exceptions.

OVR
displays the record types specified in the override record.

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Start-*ordnum*
specifies the starting set of exception ordinal numbers to be displayed, where *ordnum* is a number from 1 to FF.

Additional Information

- As many as 8 record types are displayed per line.
- In 1052 state, the display is truncated with `..T` after 44 record types are displayed because of the lack of pool records in that state.
- In NORM state, a maximum of 48 record types can be displayed. If there are more than 48 record types to be displayed, MORE is displayed on the last line.
- In either 1052 state or NORM state, to see the remaining record types, enter:

  ZDBSO DISPLAY ALL RECTYPE-*recvalue*

  where *recvalue* is the last record type that was displayed.

Examples

The record types to be captured are displayed in NORM state in the following example. Notice that only the first 48 record types in use are displayed. MORE, which is on the last line of the display, shows that there are more record types in use.

```
User:   ZDBSO DISPLAY ALL

System: DBRO0067I 10.22.21 RECORD TYPES TO BE CAPTURED
        0000 0001 0002 0003 0004 0005 0006 0007
        0008 0009 000A 000B 000C 000D 000E 000F
        0010 0011 0012 0013 0014 0015 0016 0017
        0018 0019 001A 001B 001C 001D 001E 001F
        0020 0021 0022 0023 0024 0025 0026 0027
        0028 0029 002A 002B 002C 002D 002E 002F
        MORE
```

The record types to be captured are displayed in 1052 state in the following example. Notice that only the first 44 record types in use are displayed. The ..T on the last line of the display shows that there are more record types in use.

```
User:   ZDBRO DISPLAY ALL

System: DBRO0067I 19.45.28 RECORD TYPES TO BE CAPTURED
        0005 0007 0012 0013 0014 0015 0016 0017
        0018 001A 001C 0023 0024 0025 0027 0028
        002A 002D 002E 002F 0030 0031 0032 0033
        0034 0035 0036 003A 003B 0040 0042 0043
        0044 0046 0047 0052 0059 005B 005D 0063
        0071 0073 0074 0076 0077..T
        CSMP0097I 19.45.28 CPU-B SS-BSS  SSU-HPN  IS-01
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

ZDBSO DISPLAY

In the following example, the record types to be captured are displayed in 1052 state for a subsystem with multiple subsystem users.

```
User:   WP/ZDBSO DISPLAY ALL

System: CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBRO0067I 21.45.28 RECORD TYPES TO BE CAPTURED
        0007 0011 0012 0013 0018 0019 001A 001C
        0023 0025 0027 002A 0035 0036 003A 003B
        0040 0046 0047 0052 0059 005B 0071 0073
        0076 0077 0078 0079 007A 007B 007C 007D
        007E 007F 0080 0081 0082 0083 0084 0085
        0087 0089 008A 0090 009A.T
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP2  IS-01
        DBRO0067I 21.45.28 RECORD TYPES TO BE CAPTURED
        0027 0040 0046 009A 00A1 00A2 00A5 00A8
        00AB 00AE 00AF 00B0 00B1 00B3 00B5 00BB
        00BC 00BD 00C1 00C2 00C8 00C9 00CB 00CD
        00DE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP3  IS-01
        DBRO0067I 21.45.28 RECORD TYPES TO BE CAPTURED
        0027 0040 0046 009A 00A1 00A2 00A5 00A8
        00AB 00AE 00AF 00B0 00B1 00B3 00B5 00BB
        00BC 00BD 00C1 00C2 00C8 00C9 00CA 00CB
        00CD 00DE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBSO0001I 21.45.28 MULTIPLE SSU REQUEST COMPLETED
```

The record types that contain exceptions are displayed in the following example.

```
User:   ZDBSO DISPLAY EXC

System: DBRO0068I 19.45.28 RECORD TYPES WITH EXCEPTIONS
        0005   4ST
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

The record types specified in the override record are displayed in the following example.

```
User:   ZDBSO DISPLAY OVR

System: DBRO0069I 19.45.28 RECORD TYPES IN OVERRIDE REC
        0005 0007 0012 0014 0018
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

The record types specified in the override record for a subsystem with multiple subsystem users are displayed in the following example.

```
User:   WP/ZDBSO DISPLAY OVR

System: CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBRO0069I 21.45.28 RECORD TYPES IN OVERRIDE REC
        0027 0040 00C4
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP2  IS-01
        DBRO0069I 21.45.28 RECORD TYPES IN OVERRIDE REC
        0027 0040 00C4
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP3  IS-01
        DBRO0069I 21.45.28 RECORD TYPES IN OVERRIDE REC
        0027 0040 00C4
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBSO0001I 21.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Record type0005 is displayed in the following example.

```
User:   ZDBSO DISPLAY REC-0005

System: DBSO0077I 19.45.28 EXCEPTIONS FOR RECORD TYPE 005
         START               END
        00000000000006E   0000000000000077
        0000000000000078  000000000000007B
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO IECB–Set Database Reorganization Input Phase ECB Count

Use this command to set the available entry control block (ECB) count for the input phase of database reorganization (DBR).

If the working keypoint was not initialized, a 4-KB core block is formatted and updated with the new ECB count and filed as the working keypoint. If the working keypoint was initialized, it is updated and refiled. If the input phase is running, the in-core keypoint is updated.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- You cannot enter this command if DBR is running for the input phase of fixed file records.
- Use this command carefully. ECBs are associated with frames. Increasing the number of ECBs increases the number of frames needed for DBR. See the *TPF Migration Guide: Program Update Tapes* for additional information about allocating frames for DBR.

Format

```
►►──ZDBSO IECB── ──ecbnum──────────────────────────────────────────────►◄
```

ecbnum
: is the new available ECB count, from 01–99, to use during the input phase.

Additional Information

None.

Examples

The available ECB count is set to 10 in the following example.

```
User:   ZDBSO IECB 10

System: CSMP0097I 16.57.24 CPU-B SS-BSS  SSU-HPN  IS-01
        DBSO0000I 16.57.24 REQUEST COMPLETE
        CSMP0097I 16.57.24 CPU-B SS-BSS  SSU-HPN2 IS-01
        DBSO0000I 16.57.24 REQUEST COMPLETE
        CSMP0097I 16.57.24 CPU-B SS-BSS  SSU-HPN  IS-01
        DBSO0001I 16.57.24 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO INIT–Initialize Database Reorganization Control Records

Use this command to initialize the control records used for database reorganization (DBR).

The DBR control records can be initialized to allow the capture of all fixed file and pool records in the system. Alternatively, the DBR control records can be initialized so that all the records owned by specified subsystem users are captured.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter this command before you can access the DBR control records.
- If a previous DBR run has not ended, you cannot enter this command without the BYPASS parameter. For example, if you stop a DBR run (by using the ZDBSO ABORT command) before it is completed, you must do one of the following:
  - Restart the run and let it end before you enter this command
  - Enter this command and specify YES for the BYPASS parameter.

Format

```
                      ┌─/───┐
                      │     │    ┌─ ──BYPASS=NO──┐
►►──ZDBSO INIT── ─────▼ssu──┴────┤               ├──────────────────────►◄
                                 └─ ──BYPASS=YES─┘
```

ssu
: is one or more subsystem users. This parameter captures all the records owned by the specified subsystem users. You can specify as many as 8 subsystem users.

BYPASS
: specifies whether or not to override the control bit that is set to prevent the reinitialization of control records. Specify YES to reinitialize the control records before running DBR.

  Attention: Use the BYPASS parameter with care because it causes all DBR records to be reinitialized regardless of contents. Using this parameter can cause loss of data that was previously entered by the installation. Any previous values that were set with other DBR commands are overwritten.

Additional Information

After initialization, you can use other commands associated with DBR to specify what sections of the database should not be captured during a DBR run.

Examples

In the following example, the control record is set to capture all the records owned by the SSU3, SSU4, and SSU5 subsystem users.

```
User:   ZDBSO INIT SSU3/SSU4/SSU5

System: DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO OECB–Set Database Reorganization Output Phase ECB Count

Use this command to set the available entry control block (ECB) count for the database reorganization (DBR) output phase. If the output phase is not running, the master keypoint is updated with the new available ECB count. If the output phase is running, the core copy of the working keypoint is updated to reflect the new available ECB count.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- Use this command carefully. ECBs are associated with frames. Increasing the number of ECBs increases the number of frames needed for DBR. See the *TPF Migration Guide: Program Update Tapes* for additional information about allocating frames for DBR.

Format

```
►►──ZDBSO OECB── ──ecbnum──────────────────────────────────────────────────►◄
```

ecbnum
: is the new available ECB count, from 01–99, to use during the output phase.

Additional Information

None.

Examples

The ECB count is set to 30 in the following example.

```
User:   ZDBSO OECB 30

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO RESET–Reset Database Reorganization Capture Settings

Use this command to reset the:
- Capture bit for a specific record type
- Exception ordinal number range for a specific record type
- In-use pool control bit
- DBR override record.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►──ZDBSO RESET── ──┬─RECtype-recvalue──┬────────────────────────┬─┬──────►◄
                    │                   └──RANge-startord.endord─┘ │
                    ├─DIR──────────────────────────────────────────┤
                    └─OVR──────────────────────────────────────────┘
```

RECtype-*recvalue*
: specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
: Small, long-term

SST
: Small, short-term

SDP
: Small, long-term duplicate

LLT
: Large, long-term

LST
: Large, short-term

LDP
: Large, long-term duplicate

4LT
: 4-KB, long-term

4ST
: 4-KB, short-term

4DP
: 4-KB, long-term duplicate

4D6
: 4-KB, long-term FARF6 duplicate.

RANge-*startord***.***endord*
removes the specified ordinal number range from the specified record type, where *startord* and *endord* are 1- to 16-digit hexadecimal numbers.

DIR
resets the bypass not-in-use bit to allow all pool records to be captured.

OVR
sets all entries in the DBR override keypoint to 0.

Additional Information

If you specify the RECTYPE parameter without the RANGE parameter, the capture bit for all ordinal numbers in the record type is set.

Examples

In the following example, records 110–119 in record type 0005 are reset to be captured.

```
User:   ZDBSO RESET REC-0005 RANGE-6E.77

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

In the following example, records 2–5 in record type 0027 are reset to be captured for a subsystem with multiple subsystem users.

```
User:   WP/ZDBSO RESET REC-0027 RANGE-2.5

System: CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP2  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP3  IS-01
        DBRO0000I 21.45.28 REQUEST COMPLETE
        CSMP0097I 21.45.28 CPU-B SS-WP   SSU-WP1  IS-01
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

All pool records are reset to be captured in the following example.

```
User:   ZDBSO RESET DIR

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO RESTART–Restart Database Reorganization Output Phase

Use this command to restart the database reorganization (DBR) output phase. Save the tape from the previous run for the input phase and mount a new tape when restart requests a tape.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- The TPF system must be in 1052 state unless record logging for the capture and restore utility is active. If logging is active, the system can be in NORM state when you enter this command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►──ZDBSO RESTART──┬─────────────────────┬──────────────────────────────►◄
                   └──RECtype-recvalue───┘
```

RECtype-*recvalue*
specifies the record type, where *recvalue* must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

Additional Information

- If you do not specify the RECTYPE parameter, the working keypoint is used to restart the DBR output phase with ordinal number 0 of the record type after the last record type that was successfully captured to tape.
- If you specify the RECTYPE parameter, the DBR output phase will restart at the first valid record type following the one specified. The record type specified must be the same type (fixed or pool) that was being processed when DBR was aborted.

Examples

DBR is restarted in the following example.

```
User:   ZDBSO RESTART

System: DBRO0012I 19.45.28 DBR OUTPUT PHASE RESTARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO START–Start Database Reorganization Output Phase

Use this command to start the database reorganization (DBR) output phase. You can capture the database based on the settings in the master keypoint or set up the override keypoint to capture only part of the database. The partial capture starts when you enter this command with the END parameter.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- The TPF system must be in 1052 state unless record logging for the capture and restore utility is active. If logging is active, the system can be in NORM state when you enter this command.
- In a multiple database function (MDBF) environment, you cannot enter this command for a subsystem or subsystem user that is currently running DBR.

Format

```
►►──ZDBSO START── ──┬─ALL──────────────────────┬──────────────────►◄
                    │ ┌──────────┐             │
                    │ ▼          │             │
                    └──rectype───┴─┬───────┬───┘
                                   └──END──┘
```

ALL
starts the database capture as specified by the master keypoint.

rectype
is a single record type or group of record types to be placed in the DBR override keypoint. The record type must be a 4-digit hexadecimal number for fixed file records or one of the following for pool records:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

You can specify as many as 8 record types. If more than 8 record types are required, enter the command with 8 record types and do not specify the END parameter; then enter the command again with the remaining record types followed by the END parameter. Processing does not begin until you specify the END parameter. If any previous ZDBSO START commands with the *rectype* parameter specified were entered, these records are also captured.

END
specifies that all record types to be placed in the override keypoint have been specified and starts the partial capture.

Additional Information

If you want to run the DBR output phase with buffered tape devices, you must do one of the following:

- Specify the DBRBUF=YES parameter on the CONFIG macro in your SIP stage 1 deck. See *TPF System Generation* for more information about the CONFIG macro.
- Enter the ZSYSG command with the DBRBUF parameter.

Attention: Running the DBR output phase with buffered tape devices will improve performance; however, if the system re-IPLs or tape errors occur while DBR is in progress, you may lose data. Also, because the switch that is set by the DBRBUF parameter is subsystem-shared, you can specify ZSYSG with the DBRBUF parameter only from the basic (BSS) subsystem.

Examples

In the following example, DBR is started for 4-KB long-term pool records.

```
User:   ZDBSO START 4LT END

System: DBRO0010I 07.30.32 DBR OUTPUT PHASE STARTED
```

In the following example, DBR is started for record types 0027, 0040, 0046, and 009A in a subsystem with multiple subsystem users.

```
User:   WP/ZDBSO START 0027 0040 0046

System: CSMP0097I 17.16.05 CPU-C SS-WP   SSU-WP1  IS-01
        DBRO0000I 17.16.05 REQUEST COMPLETE
        CSMP0097I 17.16.05 CPU-C SS-WP   SSU-WP2  IS-01
        DBRO0000I 17.16.05 REQUEST COMPLETE
        CSMP0097I 17.16.05 CPU-C SS-WP   SSU-WP3  IS-01
        DBRO0000I 17.16.05 REQUEST COMPLETE
        CSMP0097I 17.16.05 CPU-C SS-WP   SSU-WP1  IS-01
        DBSO0001I 17.16.05 MULTIPLE SSU REQUEST COMPLETED

User:   WP/ZDBSO START 009A END

System: CSMP0097I 17.16.05 CPU-C SS-WP   SSU-WP1  IS-01
        DBRO0010I 17.16.05 DBR OUTPUT PHASE STARTED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO STATUS–Display Database Reorganization Status

Use this command to display the following status information about the DBR run:

- Current record type
- Ordinal number
- Available entry control block (ECB) counts that are being processed by the database reorganization (DBR) output phase.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- DBR must be running when you enter this command.

Format

```
►►──ZDBSO STATUS──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

DBR status is displayed in the following example.

```
User:   ZDBSO STATUS

System: DBRO0062I 19.45.28 DBR IS PRESENTLY PROCESSING
        RECORD TYPE SDP  ,ORDINAL NUMBER 000000000000031F,ECB- 00000004
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBSO SWITCH–Restart Database Reorganization after Tape Error

Use this command to restart the database reorganization (DBR) run if an irrecoverable tape error occurs. If an irrecoverable tape error occurs, the current tape is closed, a message is sent to the console requesting a new tape, and DBR enters a defer loop. When the new tape is mounted enter this command to restart DBR activity.

Requirements and Restrictions

- You can enter this command only in a multiple database function (MDBF) environment.
- You must enter the ZDBSO INIT command before you enter any other ZDBSO command.
- You can enter this command only when you are prompted by the TPF system.
- Save the current tape for the input phase of DBR.

Format

```
►►──ZDBSO SWITCH──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

In the following example, DBR is restarted after a tape error occurred.

```
User:   ZDBSO SWITCH

System: DBRO0015I 19.45.28 DBR HAS FINISHED PROCESSING RECORD TYPE 00016
        DBRO0085A 19.45.28 TAPE ERROR..SWITCH DBF, THEN ENTER ZDBRO SWITCH
        COTC0080A 19.45.28 TCLS HPN    REMOVE DBF FROM DEVICE 426
                           VSN A00245 G0006 S0001 D38K   SL  NOBLK  NOCOMP

User:   ZTMNT DBF 428 AO BP

System: COTM0008W 19.45.28 TMNT HPN    DEVICE 428 VSN A00247
                           UNEXPIRED FILE OVERWRITTEN
        COTM0046I 19.45.28 TMNT HPN    TAPE DBF MOUNTED ON DEVICE 428
                           VSN A00247 G0007 S0001 D38K   SL  NOBLK  NOCOMP

User:   ZDBSO SWITCH

System: DBRO0000I 19.45.28 REQUEST COMPLETE
        DBSO0001I 19.45.28 MULTIPLE SSU REQUEST COMPLETED
        DBRO0015I 19.45.28 DBR HAS FINISHED PROCESSING RECORD TYPE 00018
```

Related Information

See *TPF Database Reference* for more information about DBR and for a sample problem that shows the sequence of the commands.

ZDBUG–Start and Clear the VisualAge TPF Debug Server

Use this command to do the following:

- Start the VisualAge TPF debug server.
- Stop the VisualAge TPF debug server.
- Clear the trace entries from the TPF debug registration client.
- Display the trace entries from the TPF debug registration client.

Requirements and Restrictions

- The TPF system must be in CRAS state or above to start the VisualAge TPF debug server.
- A Transmission Control Protocol/Internet Protocol (TCP/IP) offload device must have CONNECTED status.
- You should try to stop current debug sessions through normal means before entering the ZDBUG command with the CLEAR parameter specified. This command will only end current debug sessions waiting for input/output (I/O) from the workstation client. Any current debug sessions with the associated debug entry cleared but not ended will get unpredictable results.
- If you enter the ZDBUG command with the CLEAR or CLEAR and IP parameters specified while an entry is active, the WORKSTATION IP and TERMINAL addresses will display zeros.

Format

```
►►──ZDBUG── ──┬─Start──────────────────────┬──────────────────────────►◄
              ├─STOp───────────────────────┤
              ├─Clear──┬───────────────┬───┤
              │        └─ ──IP-ipaddr──┘   │
              └─DISplay── ──┬─ALL─────────┬┘
                            ├─PROGram─────┤
                            ├─TERMinal────┤
                            ├─ACTive──────┤
                            └─NONactive───┘
```

Start
: starts the VisualAge TPF debug server. Once the server is started, you can use the TPF debug registration client with the TPF system.

STOp
: stops the VisualAge TPF debug server. Once the server is stopped, you cannot use the TPF debug registration client with the TPF system.

Clear
: clears all the trace entries. Any active debug sessions that have the associated entry cleared will be ended.

IP-*ipaddr*
: specifies the Internet Protocol (IP) address of an entry, where *ipaddr* is a numeric IP address.

DISplay
: displays the trace registration information, where:

ALL
displays the trace registration information for the trace-by-program and trace-by-terminal tables.

PROGram
displays the trace registration information for the trace-by-program table.

TERMinal
displays the trace registration information for the trace-by-terminal table.

ACTive
displays the trace registration information for the trace-by-program and trace-by-terminal tables only if the entry status is active.

NONactive
displays the trace registration information for the trace-by-program and trace-by-terminal tables only if the entry status is nonactive (that is, free or in use).

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZDBUG HELP
ZDBUG ?

Examples

The VisualAge TPF debug server is started in the following example.

```
User:   ZDBUG START

System: CDBS0001I 14.53.12 VISUALAGE TPF DEBUG SERVER STARTED FOR HOST:
                           9.117.197.67 (TPF31720C) ON PORT 8000
```

The VisualAge TPF debug server is stopped in the following example.

```
User:   ZDBUG STOP
System: CDBS0011I 14.53.12 TPF DEBUGGER HAS BEEN DISABLED
```

The VisualAge TPF debug server is not stopped in the following example.

```
User:   ZDBUG STOP
System: CDBS00191I 14.53.12 AN ACTIVE TRACE ENTRY WAS FOUND. TPF DEBUGGER WAS NOT
                            DISABLED. ALL NONACTIVE TRACE ENTRIES WERE REMOVED.
```

In the following example, a single trace entry from the TPF debug registration client is removed from the VisualAge TPF debug server depending on the workstation IP address that is entered.

```
User:   ZDBUG CLEAR IP-9.117.198.100

System: CDBS0012I 14.53.12 TRACE ENTRY HAS BEEN REMOVED FOR 9.117.198.100
```

In the following example, the trace entry is cleared while the entry is active.

```
User:   ZDBUG CLEAR IP-9.117.198.100
System: CDBS0018W 14.53.12 IP ADDRESS 9.117.107.102 CLEARED WHILE ENTRY WAS ACTIVE
```

In the examples that follow, the following TPF debugger registration information is displayed:

- The TYPE field indicates whether you are using the TPF Performance Execution Trace Analyzer for VisualAge Client (PA) or the TPF Assembler Debugger for VisualAge Client or TPF C Debugger for VisualAge Client (DEBUG).
- A value of active (Act) in the STATE field indicates that you are actively debugging a program; when the debugger session ends, the STATE field changes to nonactive (Nonact). The STATE field is blank if you are using the TPF Performance Execution Trace Analyzer for VisualAge Client.
- The value in the USER TOKEN field is taken from the registration window. You can use this field to select an entry control block (ECB) to debug by associating a particular ECB with a debugger session; for instance, at the appropriate user exit, you would match the user token against information in the address space of the entry.
- The PROGRAM MASK field contains the names of the programs that are registered.

```
User:   ZDBUG DISPLAY ALL
System: CSMP0097I 17.16.04 CPU-C SS-BSS  SSU-HPN  IS-01
        CDBS0013I 17.16.04 START OF DEBUGGER REGISTRATION INFORMATION

                   TRACE BY PROGRAM TABLE
--------------------------------------------------------------------------------
WORKSTATION IP  TYPE   STATE    USER TOKEN   PROGRAM MASK
--------------------------------------------------------------------------------

9.117.107.102   Debug  Nonact                 QPK0
9.117.107.15    Pa     Nonact   BRADDK        CVZZQTA7


                   TRACE BY TERMINAL TABLE
--------------------------------------------------------------------------------
WORKSTATION IP  TYPE   TERMINAL  STATE       USER TOKEN   PROGRAM MASK
--------------------------------------------------------------------------------

9.117.107.10    Debug  010004    Nonact      BILLCO       QTA7
9.117.107.249   Debug  010005    Act         ECSMYTH      QPM3QPM9
9.117.107.102   Debug  010006    Nonact                   QPK0
9.117.107.15    Pa     010003    Nonact      BRAIN        CVZZQPK0QPM3QPM
9.117.107.15    Debug  010007    Nonact      BGOLDBRG     ABCDCHJIGARB
9.117.107.15    Pa     01000a    Nonact      KELLYK       CVZZQPM9
9.117.107.249   Debug  010008    Nonact                   QPM3QPM9
```

TPF debugger registration information is displayed in the following example only if the user entry is trace-by-program.

```
User:   ZDBUG DISPLAY PROGRAM

System: CDBS0014I 14.53.12 START OF DEBUGGER PROGRAM REGISTRATION INFORMATION

                   TRACE BY PROGRAM TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   STATE      USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   DEBUG  Nonact                  CDBS
```

TPF debugger registration information is displayed in the following example only if the user entry is trace-by-terminal.

```
User:   ZDBUG DISPLAY TERMINAL

System: CDBS0015I 14.53.12 START OF DEBUGGER TERMINAL REGISTRATION INFORMATION

                   TRACE BY TERMINAL TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   TERMINAL   STATE      USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   Debug  010000     Nonact                  CVZZ
```

In the following example, the active entries are displayed in the trace-by-terminal and trace-by-program tables.

```
User:   ZDBUG DISPLAY ACTIVE
System: CDBS0016I 14.53.12 START OF DEBUGGER ACTIVE REGISTRATION INFORMATION

                   TRACE BY PROGRAM TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   STATE      USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   Debug  Act                   QPK0

                   TRACE BY TERMINAL TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   TERMINAL   STATE   USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   Debug  010000      Act                  CDBS
```

In the following example, the nonactive entries are displayed in the trace-by-terminal and trace-by-program tables.

```
User:   ZDBUG DISPLAY NONACTIVE
System: CDBS0017I 14.53.12 START OF DEBUGGER NONACTIVE REGISTRATION INFORMATION

                    TRACE BY PROGRAM TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   STATE      USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   Debug  Nonact                  CVZZ

                    TRACE BY TERMINAL TABLE
----------------------------------------------------------------------------------
WORKSTATION IP  TYPE   STATE      USER TOKEN   PROGRAM MASK
----------------------------------------------------------------------------------

9.117.107.102   Debug  010000                  CVZZ
```

Related Information

- See "ZCLAW INACTIVATE–Deactivate CLAW Workstations" on page 215 for more information about the ZCLAW INACTIVATE command.
- See *VisualAge TPF Online Help* for more information about the VisualAge TPF debug server.

ZDCFT–Display Coupling Facility Trace Table Entries

Use this command to display entries in the coupling facility trace table (CFTT).

Requirements and Restrictions

None.

Format

```
►►──ZDCFT── ──┬─ALL─┬──────────────────────────────────────────────────►◄
              └─num─┘
```

ALL
displays all entries found in the CFTT.

num
is a decimal number from 1 to 999 that indicates the number of entries you want to display from the CFTT.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDCFT HELP
  ZDCFT ?
- The newest entries in the CFTT are displayed first, followed by the oldest entries.
- If there are fewer entries in the CFTT than the number you specified for display, all entries are displayed.

Examples

The following example shows two CFTT entries being displayed.

```
User:    ZDCFT 2

System:  DCFT0001I 07.59.14 CFDCFT - REQUESTED TRACE TABLE ENTRIES -
                            MOST RECENT FIRST
  2F82960 B0D9502C D4600205 01000001 02F829A0 *.R&;M-.&#186;.....8..*
          00001000 0102C0C0 01E0DE00 00010015 *................*
          00000000 7FFB3DFC 00000000 00000000 *...."...........*
          00000000 00000000 00000000 00000000 *................*
  2F82920 B0D9502C 4250EC07 01000001 02F82960 *.R&;.&;......8.-*
          00001000 0103C0C0 01E0CA00 00010015 *................*
          00000000 7E34F802 00000000 00000000 *....=.8.........*
          00000000 00000000 00000000 00000000 *................*
END OF REQUESTED TRACE TABLE ENTRIES
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZDCLV–Display CPU Loop and CREATE Macro Control Levels

Use this command to display the control levels for the CPU loop, the CREATE macros, and the input list. The CPU loop levels determine when to service the various lists in the loop program. The CREATE macro levels determine when to activate a CREATE macro. Main storage availability determines the levels in each case. The control levels manage the activity of the TPF system in a real-time environment.

Requirements and Restrictions

You can enter this command only in the basic subsystem (BSS).

Format

```
►►──ZDCLV──────────────────────────────────────────────────────────────────►◄
```

Additional Information

Enter the ZACLV command to change the control levels for the CPU loop, the CREATE macros, or the input list.

Examples

The following example displays the system control levels.

```
User:   ZDCLV

System: DCLV0001I 15.28.54  STORAGE CONTROLS

             DESCRIPTION     VALUE   ZACLV PARM

          SYST   MAX ECB       352     MXECBS
          INPUT  MIN FRAME     150     MNFRAI
                 MIN COMMON     18     MNCMBI
                 MIN ECB       100     MNECBI
                 MIN SWB        38     MNSWBI
                 MIN IOB        51     MNIOBI
          DEFER  MAX ECB        40     MXECBD
          TAS    MIN FRAME     300     MNFRAT
                 MIN COMMON     59     MNCMBT
                 MIN ECB       200     MNECBT
          CREM   MIN SWB        26     MNSWBM
          CRED   MIN SWB        64     MNSWBD
          CREX   MIN SWB        64     MNSWBX
          END OF DISPLAY
```

Related Information

- See *TPF Main Supervisor Reference* and *TPF Concepts and Structures* for more information about the CPU loop and the input list.
- See *TPF Main Supervisor Reference* and *TPF General Macros* for more information about the CREATE macros.

ZDCOR–Display Core

Use this command to display a variable amount of system virtual memory (SVM), also known as *core memory*.

Requirements and Restrictions

You must specify the IS parameter in a tightly coupled system when prefixing applies; for example, when you display storage in page 0 of an I-stream, or when you display the I-stream unique global areas.

Format

```
                           ┌.10─────┐   ┌ ──Is-1──┐
►►──ZDCOR── ──address──────┼────────┼───┼─────────┼───┬──────────┬──────────►◄
                           └.bytes──┘   └ ──Is-xx─┘   └ ──Instr──┘
```

address
is a 1- to 8-digit (4-byte) hexadecimal system virtual memory (SVM) address.

bytes
is the hexadecimal number of bytes to display from X'01'–X'FFF'.

Note: If you specify a value of X'00' bytes, the default number of bytes is displayed.

Is-*xx*
specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Instr
displays the data in disassembled format rather than hexadecimal format.

Note: You can display only as many as X'400' bytes at a time in disassembled format.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDCOR HELP
  ZDCOR ?
- If the address you specify is not on a fullword boundary, the display begins at the next lower fullword. If you specify the INSTR parameter, the display begins at the address that you specify.
- If the start address is in the system heap, no information is displayed if the ending address is past the end of the system heap or any part of the range is not currently in use.
- Enter the ZACOR command to change SVM.

Examples

The following example displays the prefix area of I-stream 1.

```
User:   ZDCOR 900.100

System: DCOR0010I 14.26.27 BEGIN DISPLAY
         00000900- 00000000 00000000 00000000 00000000 ........ ........
         000009F0- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays the prefix area of I-stream 2.

```
User:   ZDCOR 28.8 I-2

System: DCOR0010I 10.48.54 BEGIN DISPLAY
         00000028 - 00000010 00000194 91004140 40008400 .......m j..  .d.
        END OF DISPLAY-ZEROED LINES NOT DISPLAYED
```

The following example displays the prefix page overflowing into shared main storage.

```
User:   ZDCOR FFF IS-2

System: DCOR0010I 10.48.54 BEGIN DISPLAY
         00000FFC - 00000010 00000194 91004140 40008400 .......m j..  .d.
        END OF DISPLAY-ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about system virtual memory (SVM).

ZDCRS–Display CRAS Status

Use this command to display information from the CRAS status table.

Requirements and Restrictions

None.

Format

```
►►──ZDCRS──┬───────────┬──┬─────────┬──────────────────────────────►◄
           ├──CONS─────┤  └──cpuid──┘
           ├──lniata───┤
           ├──leid─────┤
           ├──Axx──────┤
           └──REST─────┘
```

CONS
: displays information about the functional support consoles.

lniata
: displays information about the CRAS slot for the specified address, where *lniata* is a line, interchange, and terminal address.

leid
: displays information about the CRAS slot for the specified address, where *leid* is a logical end-point identifier.

A*xx*
: displays information about the specified alternate CRAS slot, where *xx* is the number of an alternate CRAS slot from 01–99.

REST
: displays information about authorized CRAS terminals.

cpuid
: is the 1-character alphanumeric CPU ID of a processor. If you do not specify a CPU ID, information for each processor is displayed.

Additional Information

- Enter **ZDCRS** (with no parameters) to display the entire CRAS status table.
- Enter the ZACRS command to modify the contents of the CRAS status table.

Examples

The entire CRAS status table, except for the empty slots, is displayed in the following example.

```
User:   ZDCRS

System: DCRS0000I CRAS STATUS TABLE
                 TERMINAL            TERMINAL   PRINTER
                   ADRS     CPUID      TYPE      ADRS

        RO        000000      B    PRT-3284
        PRC       010000      B    CRT-3277    000000B
        RO        000000      C    PRT-3284
        PRC       010000      C    CRT-3277    000000C
        RO        000000      D    PRT-3284
        PRC       010000      D    CRT-3277    000000D
        RO        000000      E    PRT-3284
        PRC       010000      E    CRT-3277    000000E
        A01       510000      B    PRT-3284
        A02       520000      B    PRT-3284
        A03       4E0000      B    CRT-3277    510000B
        NOTE: VACANT SLOTS ARE NOT DISPLAYED
```

Information about the functional support consoles for CPU B is displayed in the following example.

```
User:   ZDCRS CONS B

System: DCRS0000I CRAS STATUS TABLE
                 TERMINAL            TERMINAL  CONSOLE
                   ADRS     CPUID      TYPE     TYPE

        RO        000000      B    PRT-3284    RO
        PRC       010000      B    CRT-3277    PRC
        DASD      000000      B    PRT-3284    DASD
        COMM      000000      B    PRT-3284    COMM
        NOTE: CONSOLES NOT ASSIGNED DEFAULT TO RO
```

Information about the terminal with address 010000 on CPU C is displayed in the following example.

```
User:   ZDCRS 010000 C

System: DCRS0000I CRAS STATUS TABLE
                TERMINAL              TERMINAL    PRINTER
                  ADRS      CPUID       TYPE       ADRS

                 010000       C      CRT-3277     000000C
```

Information about alternate CRAS number 01 is displayed in the following example.

```
User:   ZDCRS A01

System: DCRS0000I CRAS STATUS TABLE
                  TERMINAL          TERMINAL    PRINTER
                    ADRS    CPUID     TYPE       ADRS

        A01        510000     B    PRT-3284
```

Information about all terminals on CPU B with authorization to input restricted commands is displayed in the following example.

```
User:   ZDCRS REST B

System: DCRS0000I 11.11.10 CRAS STATUS TABLE
            TERMINAL         TERMINAL  AUTHORI
              ADRS   CPUID     TYPE    ZATION
        PRC  010000    B    CRT-3277   PRC
             010000    B    CRT-3277   TAPE
             010000    B    CRT-3277   DASD
             010000    B    CRT-3277   COMM
             010000    B    CRT-3277   AUDT
             010000    B    CRT-3277   RDBS
             010000    B    CRT-3277   8
             010000    B    CRT-3277   9
             010000    B    CRT-3277   10
             010000    B    CRT-3277   11
             010000    B    CRT-3277   12
             010000    B    CRT-3277   13
             010000    B    CRT-3277   14
             010000    B    CRT-3277   15
```

Related Information

See *TPF Data Communications Services Reference* for more information about CRAS support.

ZDDAT–Display Date

Use this command to display the system or subsystem date.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZDDAT──┬───────┬─────────────────────────────────────────►◄
           ├──SYS──┤
           └──ALL──┘
```

SYS
displays the system date.

ALL
displays the system date and all subsystem dates for each subsystem above 1052 state.

Additional Information

Enter **ZDDAT** (with no parameters) to display the BSS date.

Examples

The following example displays the date for the BSS.

```
User:   ZDDAT

System: DDAT0001I 13.33.39 SUBSYSTEM BSS  DATE IS 16FEB94
```

The following example displays the system date.

```
User:   ZDDAT SYS

System: DDAT0012I 13.39.10 SYSTEM DATE IS 23MAY94
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Generation* for more information about the system clocks.

ZDDCA–Display Main Storage Address of Dump Label

Use this command to do the following:

- Display the main storage addresses of a dump label
- Display main storage by specifying a dump label.

Requirements and Restrictions

You must specify the IS parameter in a tightly coupled system when prefixing applies, such as displaying storage on page 0 of an I-stream, or when displaying the I-stream unique global areas. If you do not specify the IS parameter, the default, or main I-stream, is assumed.

Format

```
                                                            ┌ ──Is-1──┐
►►──ZDDCA── ──dumplabel──┬─────────────────────────────┬────┴ ──Is-xx─┴──────────►◄
                         │     ┌─0───┐  ┌─.20────┐     │
                         └─ ───┴─rsa─┴──┴─.bytes─┴─────┘
```

dumplabel
: is a 3-character alphanumeric dump label.

rsa
: is a 1- to 8-digit hexadecimal offset.

bytes
: is the hexadecimal number of bytes to display from X'01'–X'FFF'.

Is-*xx*
: specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDDCA HELP
  ZDDCA ?
- If you do not specify a value for both *rsa* and *bytes*, the address of the dump label is displayed. To display main storage, you must specify a value for one or both of these variables.
- Not all dump tags are initialized at system startup. They are set as part of the initialization process for the facility that uses the tag. For example, SNR dump tag is not initialized until SNA is initialized. Until that point, SNR maintains a dump tag address of 0.
- Enter the ZADCA command to change main storage.

Examples

The following example displays the address of each GX1 dump label in the TPF system.

```
User:   ZDDCA GX1

System: DDCA0008I 09.12.08 DUMP TAG GX1 I-STREAM 01 ADDRESS - 01316000
                           DUMP TAG GX1 I-STREAM 02 ADDRESS - 01318000
```

The following example displays 40 bytes beginning at the address of the CLV dump label for I-stream 1.

```
User:   ZDDCA CLV .40

System: DDCA0010I 13.17.05 BEGIN DISPLAY
         00000BBC- 0096012C 0096012C 00000000 00000000 .o...o.. ........
         00000BEC- 00000000 05030704 06050000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about displaying storage using a dump label.

ZDDSI–Display Input/Output (I/O) Device Status Information

Use this command to display the status of the subchannel for an I/O device.

Requirements and Restrictions

The specified I/O device must be known to the TPF system before you enter this command.

Format

```
►►──ZDDSI DISplay── ──sda──────────────────────────────────────────────►◄
```

sda
The symbolic device address (SDA) where *sda* is the 1- to 4-digit hexadecimal symbolic address of the I/O device.

Additional Information

None.

Examples

The information for the EE5 SDA is displayed in the following example.

```
User:   ZDDSI DIS EE5

System: CDSI0001I 09.25.54 SDA 0EE5  SCH# 00010013 MOUNTED
        LDEV   013C1800
        PIM    F0   POM    FF   PNOM   00
        PAM    F0   LPM    F0   LPUM   00
        CHPID-00 18  <-- ACTIVE
        CHPID-01 26  <-- ACTIVE
        CHPID-02 99  <-- ACTIVE
        CHPID-03 A5  <-- ACTIVE
```

The information for the 0194 SDA is displayed in the following example.

```
User:   ZDDSI DIS 0194

System: CDSI0002I 12.48.05 SDA 0194 NOT DEFINED
```

The information for the FFFF SDA is displayed in the following example.

```
User:   ZDDSI DIS FFFF

System: CDSI0003I 12.48.05 SDA FFFF OUT OF RANGE.  HIGHEST SDA IS 1FFF
```

Related Information

See *ESA/390 Principles of Operation* for more information about PIM, PAM, and POM.

ZDEAT–Display the ECB Activation Table (EAT)

Use this command to display the ECB activation table. The ECB activation table contains the following information:

- Activation numbers in increasing order
- Number of ECBs using each activation number
- Whether the activation number corresponds to a selectively activated loadset.

Requirements and Restrictions

None.

Format

```
►►──ZDEAT──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example displays the ECB activation table (EAT).

```
User:   ZDEAT

System: DEAT0001I DISPLAY ECB ACTIVATION TABLE
         ACTIVATION NUMBER       ECB COUNT    SELECTIVELY ACTIVATED
                         1               3                        N
                         2               0                        Y
                         5             204                        N
                         6              15                        Y
                         7             538                        N
        END OF DISPLAY
```

Related Information

None.

ZDEBB–Display Blocked Tapes Online

Use this command to display *blocked tapes* online. You can:
- Display records on a blocked tape
- Shift a blocked tape forward or backward
- Search a blocked tape for a specific item.

A *blocked tape* is a tape of physical records that can consist of one or more contiguous logical records or parts of logical records. Physically, the blocked tape is read one record at a time; logically, the TPF system manages the blocked tape record by record.

Requirements and Restrictions

- The specified blocked tape must be mounted as a DBL active input tape using the ZTMNT command. A DBL tape label must exist before entering the ZDEBB command.
- The ZDEBB command can be used only to process a single tape. If ZDEBB command processing encounters a tape mark with an end-of-volume (EOV) label on the DBL tape, the TPF system treats the EOV condition as an end-of-file (EOF) condition and does not attempt to switch to a standby DBL tape. To process multiple blocked tapes in a dataset, use the ZDEBB command separately for each tape.
- The display and print tape block parameters (FD, FP, RD, RP, SD, SP, TD, and TP) require that the TPF system is in a state where the get file storage (GFS) facility is active (that is, computer room agent set (CRAS) state or higher). In addition, the TPF system must not be cycling to 1052 state. For the SD and SP parameters, the tape search takes place before checking these constraints and displaying the tape information.
- After displaying logical records on the blocked tape or searching for information in a logical record, enter the ZDEBB command with the RU parameter specified to do the following:
  – Rewind and unload the blocked tape
  – Remove the DBL entry from the tape status table.

Format

```
►►──ZDEBB── ──┬─RU──────────────────────────────────────────┬────────────►◄
              ├─RW──────────────────────────────────────────┤
              ├─SF──────────────────────────────────────────┤
              ├─TD──────────────────────────────────────────┤
              ├─TP──────────────────────────────────────────┤
              ├─BB── ──blocks───────────────────────────────┤
              ├──┬─FD─┬── ──records─────────────────────────┤
              │  ├─FP─┤                                     │
              │  ├─RD─┤                                     │
              │  ├─RP─┤                                     │
              │  └─SR─┘                                     │
              └──┬─SD─┬── ──displacement── ──┬─C─┬──-data-──┘
                 ├─SP─┤                      └─X─┘
                 └─TS─┘
```

RU
: rewinds and unloads the blocked tape. If this is not done, the DBL tape will remain mounted.

RW
: rewinds the blocked tape.

SF
: shifts the blocked tape forward one or more logical records.

TD
: performs a tape block hexadecimal dump of the next logical record.

TP
: prints the next logical record in character format.

BB
: shifts the blocked tape backward the specified number of physical blocks.

blocks
: is the number of physical blocks from 0000–9999.

FD
: shifts the blocked tape forward the specified number of logical records and performs a tape block hexadecimal dump (SR and TD).

FP
: shifts the blocked tape forward the specified number of logical records and prints the tape block in character format (SR and TP).

RD
: shifts the blocked tape backward the specified number of logical records and performs a tape block hexadecimal dump (BB and TD).

RP
: shifts the blocked tape backward the specified number of logical records and prints the tape block in character format (BB and TP).

SR
: shifts the blocked tape forward the specified number of logical records.

records
is the number of logical records from 0000–9999.

SD
searches the blocked tape for the specified data and performs a tape block hexadecimal dump (TS and TD).

SP
searches the blocked tape for the specified data and prints the tape block in character format (TS and TP).

TS
searches the blocked tape for the specified data.

displacement
is the 3-byte hexadecimal displacement in the logical record.

C indicates that the data is character data.

X indicates that the data is hexadecimal data.

-*data***-**
is the character or hexadecimal data.

Note: You must begin and end the data with the - character.

Additional Information

None.

Examples

The blocked tape is rewound in the following example.

```
User:   ZDEBB RW

System: CSAV0000I 11.44.53 OPERATION COMPLETE
```

The next tape block is displayed in the following example.

```
User:   ZDEBB TP

System: CSAW0000I 11.44.53 TAPE DATA RECORD DISPLAY
         000  T D 1 0  5 5   B  L K 0 0  0 0 0 1
         010  9 9 900 00000000 00000000 00000000
         410 5C E N D    O F    R E C O  R D5C O
         END OF DISPLAY - RECORD LENGTH IS  1055
```

The blocked tape is shifted forward 1 block in the following example.

```
User:   ZDEBB SR 0001

System: CSAV0000I 14.45.34 OPERATION COMPLETE
```

The blocked tape is shifted backward one physical block and the logical record is printed in character format in the following example.

```
User:   ZDEBB RP 0001

System: CSAW0000I 11.44.53 TAPE DATA RECORD DISPLAY
         000  T D 1 0  5 5   B  L K 0 0  0 0 0 2
         010  0 0 000 00000000 00000000 00000000
         410 5C E N D    O F    R E C O  R D5C O
         END OF DISPLAY - RECORD LENGTH IS  1055
```

Related Information

None.

ZDEBE–Display and Control Blocked and Unblocked Tapes Online

Use this command to examine *blocked* and *unblocked tapes* online. You can:
- Shift a blocked or unblocked tape forward or backward
- Search a blocked or unblocked tape for a specific item
- Display blocked or unblocked tape data blocks.

A *blocked tape* is a tape of physical records that can consist of one or more contiguous logical records or parts of logical records. Physically, the blocked tape is read one record at a time; logically, the TPF system manages the blocked tape record by record.

An *unblocked tape* contains one physical record for each block.

Requirements and Restrictions

- The ZDEBE command cannot be used when working with logical records in a blocked tape. Use the ZDEBB command instead.
- The specified tape device must be free for use.
- The display and print tape block parameters (BD, BP, FD, FP, SD, SP, TD, and TP) require that the TPF system is in a state where the get file storage (GFS) facility is active (that is, CRAS state or higher). In addition, the TPF system must not be cycling to 1052 state. For the SD and SP parameters, the tape search takes place before checking these constraints and displaying the tape information.
- All forward and backward tape shifting and tape search operations end immediately if a tapemark is detected.

Format

```
►►──ZDEBE── ──cuu── ──┬─BF──────────────────────────────────────┬──────────►◄
                      ├─RU──────────────────────────────────────┤
                      ├─RW──────────────────────────────────────┤
                      ├─SF──────────────────────────────────────┤
                      ├─TD──────────────────────────────────────┤
                      ├─TL──────────────────────────────────────┤
                      ├─TP──────────────────────────────────────┤
                      ├─WT──────────────────────────────────────┤
                      ├──┬─BD─┬─ ──blocks───────────────────────┤
                      │  ├─BP─┤                                 │
                      │  ├─BS─┤                                 │
                      │  ├─FD─┤                                 │
                      │  ├─FP─┤                                 │
                      │  └─SR─┘                                 │
                      └──┬─SD─┬─ ──displacement── ──┬─C─┬─-data-┘
                         ├─SP─┤                     └─X─┘
                         └─TS─┘
```

cuu
is a 3-digit hexadecimal tape device address.

BF
shifts the blocked or unblocked tape backward to the preceding tapemark.

RU
rewinds and unloads the unblocked tape.

Note: The blocked or unblocked tape is unloaded regardless of whether the TPF system is in test mode.

RW
rewinds the blocked or unblocked tape.

SF
shifts the blocked or unblocked tape forward.

TD
performs a tape block hexadecimal dump.

TL displays the tape label.

Note: This parameter positions the tape following the HDR2 label.

TP
prints the next tape block in character format.

WT
writes the tapemarks.

Note: The blocked or unblocked tape must be write-enabled. You cannot specify this parameter if the tape is at load point.

BD
shifts the blocked or unblocked tape backward the specified number of blocks and performs a tape block hexadecimal dump (BS and TD).

BP
shifts the blocked or unblocked tape backward the specified number of blocks and prints the tape block in character format (BS and TP).

BS
shifts the blocked or unblocked tape backward the specified number of blocks.

FD
shifts the blocked or unblocked tape forward the specified number of blocks and performs a tape block hexadecimal dump (SR and TD).

FP
shifts the blocked or unblocked tape forward the specified number of blocks and prints the tape block in character format (SR and TP).

SR
shifts the blocked or unblocked tape forward the specified number of blocks.

SD
searches the blocked or unblocked tape for the specified data and performs a tape block hexadecimal dump (TS and TD)

SP
searches the blocked or unblocked tape for the specified data and prints the tape block in character format (TS and TP)

TS
searches the blocked or unblocked tape for the specified data.

blocks
is the number of blocks from 0000–9999.

displacement
is the 3-byte hexadecimal displacement in the tape block.

C indicates that the data is character data.

X indicates that the data is hexadecimal data.

-*data***-**
is the character or hexadecimal data.

Note: You must begin and end the data with the - character.

Additional Information

- The blocked or unblocked tape is internally mounted as a DBE tape and the tape status table (TSTB) entry remains seized for the duration of the operation.
- Only the WT parameter is checked for file protect status.
- If an HDR1 or HDR2 label is not found, asterisks are displayed in the related fields of the output. Similarly, asterisks are displayed in the SSN and SSU fields if the subsystem name that is contained in the HDR1 label is not recognized by the current system configuration.
- The display and print tape block parameters (BD, BP, FD, FP, SD, SP, TD, and TP) process the tape as an unblocked tape. That is, each physical block is considered one logical record.
- Hexadecimal digits are converted to printable characters when possible. Otherwise, the hexadecimal representation of the data is displayed.
- Only the first 4-KB of a tape block is displayed at one time.
- If a 16-byte line of data contains all X'00' characters, the line is not displayed.
- After data is displayed, the tape is positioned before the next block.

Examples

The tape label for the specified tape device is displayed in the following example, where:

VOLUME SERIAL
is the volume serial number, which should be the same as the external label on the unblocked tape.

SSN
is the name of the subsystem in which the unblocked tape was created.

SSU
is the name of the subsystem user in which the unblocked tape was created.

F is the name of the data set.

G is the generation number of the unblocked tape.

S is the volume sequence number of the unblocked tape.

FILE SERIAL NUMBER
is the file serial number of the first unblocked tape in the data set.

CREATION DATE
is the date the unblocked tape was created.

EXPIRATION DATE
is the date the unblocked tape expires and can be written over.

BLOCKING FORMAT
is the blocking mode, either BLK or NOBLK.

COMPACTION MODE
is the improved data recording capability (IDRC) mode, which can be COMP (compaction) or NOCOMP (no compaction).

Notice that asterisks are displayed in the SSN and SSU fields. This indicates that the subsystem name that is contained in the HDR1 label is not recognized by the current system configuration.

```
User:   ZDEBE 480 TL

System: COTP0001I 09.59.05 DEBE      - TAPE LABEL RECORD DISPLAY
             VOLUME SERIAL  -A00136-  SSN-****    SSU-****
             F-VIRTUAL.TAPE           G  -0001    S  -0001
             FILE SERIAL NUMBER    -  A00136
             CREATION DATE   - 00000   BLOCKING FORMAT - NOBLK
             EXPIRATION DATE - 00000   COMPACTION MODE - NOCOMP
```

The next tape block is displayed in the following example.

```
User:   ZDEBE 482 TP

System: COTQ0001I 12.32.58 DEBE      - TAPE DATA RECORD DISPLAY
        000 00800000 007C0000  D G     00000000
        010 A991B92C 132B1150 00000065 00000000
        070  B S S    H P N   A991B92C 132BB150
        END OF DISPLAY - RECORD LENGTH IS   128
        ZEROED LINES SUPPRESSED
```

The unblocked tape is shifted forward 2 blocks in the following example.

```
User:   ZDEBE 482 SR 0002

System: COTO0044I 12.06.51 DEBE HPN    COMPLETE
```

In the following example, the unblocked tape is searched for the first block that contains the VOL1 characters.

```
User:   ZDEBE 482 SD 000 C-VOL1-

System: COTQ0001I 12.08.43 DEBE      - TAPE DATA RECORD DISPLAY
        000 E5D6D3F1 C1F0F0F1 F4F5F040 40404040
        010 40404040 40404040 40404040 40404040
        020 40404040 40404040 40E5E3C1 D7C540E5
        030 E2E2C940 40404040 40404040 40404040
        040 40404040 40404040 40404040 40404040
        END OF DISPLAY - RECORD LENGTH IS    80
```

Related Information

See “ZDEBB–Display Blocked Tapes Online” on page 303 for more information about the ZDEBB command.

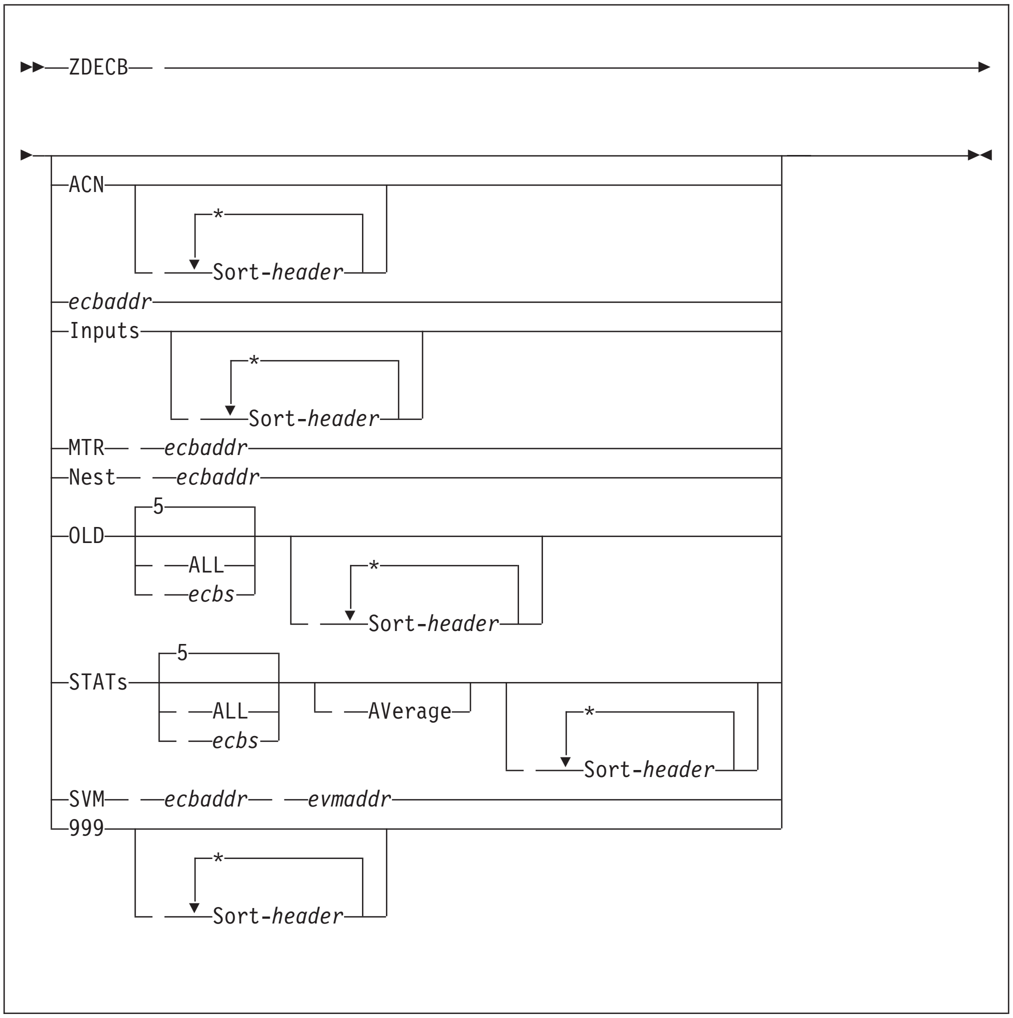
ZDECB–Display In-Use Entry Control Blocks

Use this command to display information about entry control blocks (ECBs) that are in use.

Requirements and Restrictions

None.

Format



ACN
: displays ECBs by activation number.

Sort-*header*
: displays ECBs sorted by input header, where *header* is one of the following:

  ACN
  : The ECB activation number.

  ADDR
  : The system virtual memory (SVM) address.

  DSP
  : The displacement of the last macro issued in a program.

FILE
The collection of FILE type macros issued.

FIND
The collection of FIND type macros issued.

FRM
The frames attached to an ECB.

GETF
The GETFC macro issued.

HOLD
The ECB records held.

INPUT
The ECB input message.

IO The status of I/O operations.

IS The I-stream.

MILS
The amount of CPU time, in milliseconds, used by the ECB.

ORIGIN
The information about the ECB that is created.

PGM
The program name.

SSU
The subsystem user (SSU).

SVC
The last supervisor call (SVC) issued.

TIME
The amount of time an ECB has been active.

You can specify as many as three input headers.

ecbaddr
displays detailed ECB information, where *ecbaddr* is the SVM ECB address. This information includes the I/O count, records held, and the chain addresses of entries waiting for records held.

Inputs
displays all ECBs that are in use and input messages that result from the ECBs being created. The display is sorted by time and shown in minutes and seconds.

Note: You must modify the CVXS segment to display the input message field.

MTR *ecbaddr*
displays a macro trace for an ECB that is in use, where *ecbaddr* is the ECB address.

Nest *ecbaddr*
displays nesting level information for an ECB, where *ecbaddr* is the ECB address.

OLD
displays information about the specified number of active ECBs and the total number of ECBs that are in use in the TPF system. If you do not specify a

number, the default is the five oldest active ECBs are displayed. The display is sorted by time, starting with the oldest ECB, and shown in minutes and seconds.

Note: If you do not enter any parameters with the ZDECB command, it is equivalent to entering the ZDECB command with the OLD parameter specified.

ALL
displays all ECBs that are in use.

ecbs
is the number of ECBs to be displayed.

STATs
displays statistics for the specified number of ECBs that are in use with the highest total of milliseconds (ms). If you do not specify a number, the default is five ECBs in use are displayed.

AVerage
displays the average count per second of the GETFC, FIND, and FILE macros for the specified number of ECBs that are in use, and is sorted by milliseconds per second.

SVM *ecbaddr evmaddr*
displays the equivalent SVM address for the ECB virtual memory (EVM) address specified, where *ecbaddr* is the SVM ECB address and *evmaddr* is the EVM address.

999
displays ECBs older than or equal to the number of seconds specified in the range 0–999.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZDECB HELP
ZDECB ?

Examples

The following example displays all ECBs that are in use and equal to or older than 3 seconds. The asterisks (*) indicate the ECBs that are not active during the time frame specified. File addresses that are in 4x4 mode will be displayed as 4-byte addresses. File addresses that are 8-byte FARF6 addresses will not be displayed so that the line will not wrap.

```
User:   ZDECB 3

System: DECB0014I 11.45.38 DISPLAY ECB SUMMARY
        ECB ADDR  SSU IS  PGM   MIN SC  ORIGIN  I H DSP SVC
        01A0C000 HPN   1 CVSNTS  63 26  1052    1       FILNC CC022801
        01A30000 HPN   1 CVSNTS  63 23  1052    1       FINDC CC022801
        01A42000 HPN   1 CVSNTS  63 20  1052    1   1FC FINWC CC022801
        01A4E000*HPN   1 CVSNTS  63 18  1052    1   2CC TOPNC
        01D39000*HPN   1 CVSNTS     5   1052    1 2 536 FIWHC CC022804
        01D51000*HPN   1 CVSNTS     5 CREM CVSN 1 2 620 FIWHC CC022803
        01DFF000 HPN   1 CLTW40 63 30 CREM CLTY 1   DLM EVNWC
        TOTAL     7
        END OF DISPLAY
```

In the following example, the nesting level information for an ECB is displayed.

```
User:   ZDECB NEST 1B88000

System: DECB0011I 07.15.00 PROGRAM NESTING FOR ECB 01B88000 STARTED
        ECB ADDR  SSU IS  PGM   MIN SC  ORIGIN   I H DSP  SVC
        01B88000 HPN   1 COSA40   8 58   CXFR    1 1 2E8  EVNWC
           MILS  FRM   FIND   FILE   GETF  INPUT
            987   11     43      7      0
        NESTING LEVEL  PROGRAM NAME  DISPLACEMENT
              1          COTB40         16A
              0          CTKS40         40C
        END OF DISPLAY
```

In the following example, statistics for the five in-use ECBs with the highest total of milliseconds are displayed under MILS.

```
User:   ZDECB STAT

System: DECB0014I 11.49.35 DISPLAY ECB SUMMARY
        ECB ADDR IS PGM    MIN SC  MILS  FRM   FIND    FILE  GETF
        01A0C000  1 CVSNTS  67 23 550356   8      2  772505     0
        01A30000  1 CVSNTS  67 20 439617   8 770727       0     0
        01A42000  1 CVSNTS  67 17 402578   8 770030       0     0
        01DFF000  1 CLTW40  67 27   1739   7      0       0     0
        01AC0000  1 CVSNTS      4     25   8      3       0     0
        TOTAL    8
        END OF DISPLAY
```

In the following example, the record held information of the specified ECB is displayed.

```
User:   ZDECB 0263A000

System: DECB0012I 15.27.49 ANALYZER FOR ECB 0263A000 STARTED
        ECB ADDR  SSU IS  PGM   MIN SC  ORIGIN   I H DSP  SVC
        0263A000 HPN   1 CVSNTS     23   1052    1 2 53A FIWHC F4023811
            MILS  FRM   FIND   FILE   GETF  INPUT
               8   10      3      0      0

        WAITING ON - FIWHC 00000000F4023811

        RECORD HOLD TABLE INFORMATION FOR ECB 0263A000
          HOLDING F/A      WAITING F/A    HOLDING ECB
                         00000000F4023811  02640000
        00000000F402380D
        ECB ANALZYER COMPLETE FOR ECB 0263A000
```

In the following example, the macro trace information of the specified ECB is displayed.

```
User:   ZDECB MTR 2433000

System: DECB0013I 15.50.18 MACRO TRACE FOR ECB 02433000 STARTED
        ECB ADDR  SSU IS  PGM   MIN SC  ORIGIN   I H DSP  SVC
        02433000 HPN   1 CVSNTS     17   1052    1   206 FIWHC F4023805
            MILS  FRM   FIND   FILE   GETF  INPUT
            1487   10   8870      0      0
        START OF MACRO TRACE FOR ECB 02433000
        MACRO                  IS   PROGRAM PSW    PROG      TIME

        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A1DC8C105
        RELCC                  01 071D00008373320E CVSN B3D71D2A1E7D9385 _
        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A1E7DBB85
        RELCC                  01 071D00008373320E CVSN B3D71D2A1F9E4D46
        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A1F9E8046
        RELCC                  01 071D00008373320E CVSN B3D71D2A2057CE07
        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A2057FA07 _
        RELCC                  01 071D00008373320E CVSN B3D71D2A2111C448
        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A2111EE88
        RELCC                  01 071D00008373320E CVSN B3D71D2A21CA7840
        FINWC 00000000F4023805 01 071D000083733206 CVSN B3D71D2A21CAA740
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE+
```

In the following example, the EVM address of the specified ECB is converted to an SVM address. The EVM address used in this example is the address of a core block attached to the ECB. The EVM address was obtained by entering the ZDCOR command to display the ECB.

```
User:   ZDECB SVM 1A42000 61EE80

System: DECB0005I 11.51.14 ECB- 1A42000, EVM-61EE80, SVM-84EE80
```

Related Information

None.

ZDECD–Set or Display Address Translation Dump

Use this command to set or display dump switches for a record. These dump switches are contained in the FACE table and are set for specific record type uniqueness groups.

When a dump switch is set for a record type uniqueness group, a SNAPC dump is taken if a FARF address is specified that is not in the format currently being dispensed

Requirements and Restrictions

None.

Format

```
►►──ZDECD── ──┬─DISplay─┬── ──┬─RECId-rectype──┬──┬──────────┬──────────►
              ├─DUmp────┤     └─RECVal-recvalue┘  └──SSU-user┘
              └─NODump──┘

►──┬──────────┬──┬──────────────┬─────────────────────────────────────►◄
   └──CPUid-id┘  └──Istream-xx──┘
```

DISplay
displays the dump switch status for the specified record type.

DUmp
sets the dump switch on for the specified record type.

NODump
sets the dump switch off for the specified record type.

RECId-*rectype*
sets or displays the dump switches for the specified record type, where *rectype* is a 2- to 8-character alphanumeric record type that can also include the number sign (#).

RECVal-*recvalue*
sets or displays the dump switches for the specified record value, where *recvalue* is a 4-digit hexadecimal record value type.

SSU-*user*
sets or displays the dump switches for the specified subsystem user, where *user* is a 1- to 4-character alphanumeric subsystem user name.

CPUid-*id*
sets or displays the dump switches for the specified processor, where *id* is a 1-character alphanumeric CPU ID.

Istream-*xx*
sets or displays the dump switches for the specified I-stream, where *xx* is a decimal number from 1 to 16.

Additional Information

None.

Examples

The dump switch is set for record type FRED in the following example.

```
User:   ZDECD DUMP RECID-#FRED

System: DECD0001I 10.48.54 DUMP SWITCH TURNED ON
```

The status of the dump switch for record type FRED is displayed in the following example.

```
User:   ZDECD DISPLAY RECID-#FRED

System: DECD0003I 10.48.54 DUMP SWITCH STATUS - MIXED
```

Related Information

See *TPF Database Reference* for more information about FARF addressing.

ZDFAI–Display File Address Information

Use this command to display information about a file address.

Requirements and Restrictions

None.

Format

```
►►──ZDFAI── ──fileaddr──────────────────────────────────────────────►◄
```

fileaddr
is an 8- or 16-digit hexadecimal file address.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZDFAI HELP
ZDFAI ?

Examples

In the following example, information about a fixed file address is displayed, where:

RECORD TYPE
is the 8-byte character representation of the record type. This value is left-justified and padded with blanks.

RECORD ORDINAL
is the 8-byte ordinal number of this record type.

RECORD STATUS
indicates whether or not the record is shared by all subsystem users, processors, and I-streams. One of the following is displayed:

- SHARED
- UNIQUE

```
User:   ZDFAI F4024011

System: CFAI0002I 16.56.13 DISPLAY FOR FIXED RECORD BEGINS
        RECORD TYPE:     #KEYPT
        RECORD ORDINAL:  0000000000000004
        RECORD STATUS:   SHARED
        END OF DISPLAY
```

In the following example, information about a pool address is displayed, where:

POOL SECTION
is the 4-byte character pool section name.

POOL ORDINAL
is the 8-byte hexadecimal ordinal number of this pool record.

DIRECTORY ORDINAL
is the 4-byte hexadecimal ordinal number of the directory for this pool record.

DIRECTORY BYTE
: is the 2-byte hexadecimal number for the byte that this pool record occupies in the directory.

DIRECTORY BIT
: is the 1-byte hexadecimal mask value for the bit that this pool record occupies in the directory.

POOL STATUS
: indicates the status of the pool record. One of the following is displayed:
- AVAILABLE
- IN-USE
- UNKNOWN

If UNKNOWN is displayed, the directory is currently in use by a processor so that this file record may be displayed as erroneously available. The processor ID of the owner will be displayed, if possible.

```
User:   ZDFAI 00000000000CD259

System: CFAI0003I 16.56.13 DISPLAY FOR POOL RECORD BEGINS
        POOL SECTION:       LDPA
        POOL ORDINAL:       0000000000003496
        DIRECTORY ORDINAL:  0000005D
        DIRECTORY BYTE:     01F1
        DIRECTORY BIT:      20
        POOL STATUS:        AVAILABLE
        END OF DISPLAY
```

In the following example, information about a pool address that resides in a pool section that is not active is displayed.

```
User:   ZDFAI 0000020040000000

System: CFAI0004I 16.56.13 DISPLAY FOR INACTIVE POOL SECTION RECORD BEGINS
        POOL SECTION:       4D6A
        POOL ORDINAL:       0000000000000080
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file address formats.

ZDFCT–Display File Address Compute Program (FACE) Table Data

Use this command to display the characteristics of a record type as well as file address and module, cylinder, head and record (MCHR) information for each extent of a record type in the file address compute program (FACE) table (FCTB).

The *FCTB* is the main storage resident table used by the FACE program to calculate the address associated with a specific record type and ordinal number on fixed file storage, and contains the following information:

- Record type
- Ordinal ranges, MCHR, and file addresses for all extents of the record type
- Attribute information such as the size of the fixed record, whether the fixed record is or is not a duplicate, and the device type the extent is on.

Requirements and Restrictions

- This command will not display information for record types that are processor unique, subsystem user (SSU) unique, or I-stream unique.
- This command will not display information for pool record types.

Format

```
►►──ZDFCT── ──rtype──────────────────────────────────────────────►◄
```

rtype
: is the 1- to 5-digit decimal record type.

Additional Information

None.

Examples

The following example displays information about the FACE table. In this example:

TYPE
: is the record type.

ORDINAL RANGE
: is the ordinal range from 0–99 999 999.

ATTRIBUTES
: is *s/d* DEV*x*, where:

  s is the record size, where:

  - S is for 381-byte.
  - L is for 1055-byte.
  - 4 is for 4095-byte.

  d is an indication of record duplication, where:

  - N is for non-duplicated.
  - D is for duplicated.

DEV*x*
is the device type (A through D) for which the fixed records for the entry are resident.

BASE FA/MMMMCCCCHHHHRR
is the following:

BASE FA
is the FARF address of the fixed record in the ordinal range.

MMMMCCCCHHHHRR
is a 14-digit MCHR address of the fixed record, where:

MMMM
is the 4-digit hexadecimal symbolic module number.

CCCC
is the 4-digit hexadecimal cylinder number.

HHHH
is the 4-digit hexadecimal head number.

RR
is the 2-digit hexadecimal record number.

```
User:   ZDFCT 20

System: CDFT0012I DISPLAY FACE TABLE DATA
        TYPE    ORDINAL RANGE      ATTRIBUTES  BASE FA/MMMMCCCCHHHHRR
        00212   00000000-00000019   4/D DEVA   F403D001/00470079000104
                00000020-00000163   4/D DEVA   F403D051/0047007B000901
                00000164-00000199   4/D DEVA   F403D291/00470181000A01
        END OF DISPLAY
```

Related Information

None.

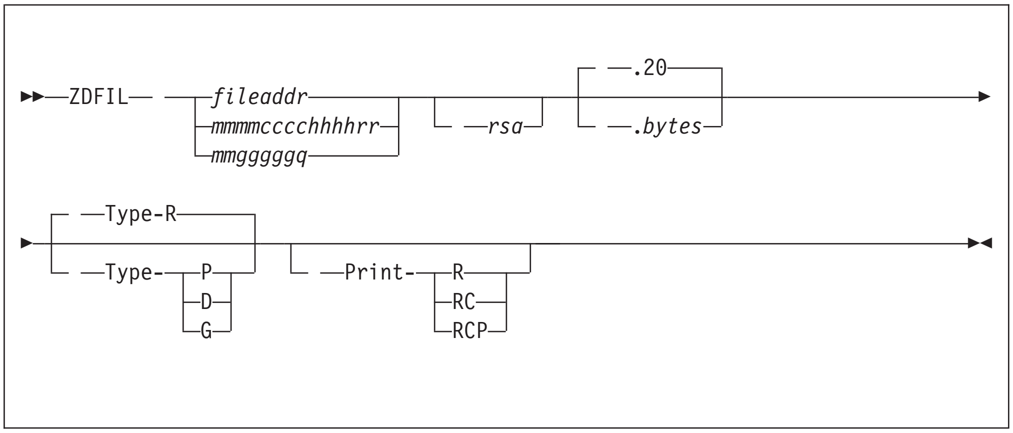
ZDFIL–Display File

Use this command to display a file record.

Requirements and Restrictions

You cannot use this command to display general dataset records.

Format



fileaddr
is an 8- or 16-digit hexadecimal file address.

mmmmccccchhhhrr
is a 14-digit extended MCHR file address, where:

mmmm
is the 4-digit hexadecimal symbolic module number.

cccc
is the 4-digit hexadecimal cylinder number.

hhhh
is the 4-digit hexadecimal head number.

rr is the 2-digit hexadecimal record number.

mmgggggq
is an 8-digit general file (GF) pseudo module number and relative record number, where:

mm
is the 2-digit hexadecimal symbolic GF module number.

ggggg
represents bits 8–27 of a GF relative record number as a binary counter.

q is a single hexadecimal digit that represents the low-order 4 bits of the GF relative record number as follows:

Bit	**Description**
28	Unit position of the relative record number.
29	Must be 1 to indicate a relative record number.
30	Must be 0.

31 The number 1 to indicate a large record, or the value 0 to indicate a small record.

rsa
is a 1- to 3-digit hexadecimal relative starting address (offset).

Note: If the relative starting address is not on a fullword boundary, it is automatically adjusted to the next lower fullword boundary.

bytes
is the hexadecimal number of bytes to display from X'01'–X'FFF'.

Note: If the number of bytes you specify does not correspond with an even number of lines, the number is automatically adjusted to display the next higher even number of lines.

Type
specifies which file copy of the record to display:

R displays the prime copy or the duplicate copy.

P displays the prime copy.

D displays the duplicate copy.

G displays the 4-byte pseudo module number and relative record number copy.

Note: The P and D options are valid only for FARF format addresses.

Print
prints the following data:

R prints to an available printer the record at the specified address.

RC
prints to the current display device the chain of addresses from the specified record for a total of 33 chains.

RCP
prints to an available printer the chain of records from the specified record, up to a total of 33 chains.

Note: The PRINT parameter is valid only for FARF3, FARF4, or FARF5 file addresses.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDFIL HELP
  ZDFIL ?
- Enter the ZCSON command to convert MCHR addresses to FARF3, FARF4, FARF5, or FARF6 format depending on the system migration stage.

Examples

The prime copy of a file record is displayed in the following example.

```
User:   ZDFIL 0000000038880007 2.40 TYPE-P

System: DFIL0011I 07.10.02 DISPLAY OF FILE ADDRESS 0000000038880007
         00000000- C1D60080 C3C7E3F2 00000514 00000515 AO..CGT2 ........
         00000010- 00020FE8 0000432C 00004EB4 000023D0 ...Y.... ........
         00000020- 00002538 00000000 00000000 00000000 ........ ........
         00000030- 00000000 00000000 00002A24 00003564 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

A general file record is displayed in the following example.

```
User:   ZDFIL 0D000005 2.40 TYPE-G

System: DFIL0011I 07.40.37 DISPLAY OF FILE ADDRESS 000000000D000005
         00000000- C3D40020 C4E8C4C7 00000000 00404040 CM..DYDG .....
         00000010- 0D000085 0D00033D 0D0005F5 00000000 ...e.... ...5....
         00000020- 00000000 004E0000 40000000 C2E2E240 ........  ...BSS
         00000030- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about displaying file records.

ZDFPC–Display File Pool Counts

Use this command to display the number of available file pool addresses.

Requirements and Restrictions

You can enter this command in 1052 state without getting an error, but nothing will happen until you cycle the system to NORM state.

Format

```
►►──ZDFPC──┬────────────────┬──────────────────────────────────►◄
           ├──DEV── ──device─┤
           ├──POOL── ──type──┤
           ├──TOTALS─────────┤
           ├──D── ──time─────┤
           └──STOP───────────┘
```

DEV *device*
displays all the pool counts for the specified device, where *device* is the name of a device.

POOL *type*
displays all the device counts for the specified pool type, where *type* is one of the following pool types:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

TOTALS
displays the total of all the device counts for each pool type.

D *time*
displays the pool counts on a time-initiated basis, where *time* is a time interval in minutes.

STOP
stops the time-initiated display.

Additional Information

- Enter ZDFPC (without any parameters) to display the file pool counts for all the device types that you have defined. You can define as many as four device types.
- In all displays except ZDFPC TOTALS, the active directory ordinal is displayed in hexadecimal format under keyword ORD.
- The format of the count display is different depending on the system status.

  For example, when the TPF system is in 1052 state, the display shows only the directory counts that are available and are not owned. Short-term directories that aged sufficiently are counted. When the TPF system is in NORM state, the resources that are not owned are displayed with the FILE indicator. The keyword CORE indicates the resources owned by the requesting processor.
- If the pool monitor is on and the TPF system is above UTIL state when you enter this command, the number of pool records from the designated long-term pool section that were used in the last minute is also displayed under the keyword RATE.

Examples

The device counts for large, long-term duplicate pool records are displayed in the following example.

Note: The TPF system is in 1052 state.

```
User:   ZDFPC POOL 4DP

System: DFPC0011I 16.35.34 26MAR DFPC  AVAILABLE FILE POOL COUNTS
                                     FILE      ORD
        4DP DEVA                  104 772 0000007D
            DEVB                    4 639 000000B2
            TOT                   109 411
        END OF DISPLAY
```

The total of all the device counts for each pool type is displayed in the following example.

Note: The TPF system is in NORM state.

```
User:   ZDFPC TOTALS

System: DFPC0011I 08.22.01 15FEB DFPC  AVAILABLE FILE POOL COUNTS
                                     FILE      CORE      RATE
        SST TOT                   189 750       875
        SDP TOT                   138 052       550         0
        LST TOT                    68 838         0
        LDP TOT                    59 160     14890        10
        4ST TOT                    34 176         0
        4DP TOT                   109 157         0         0
        4D6 TOT                    14 036     14900         0
        END OF DISPLAY
```

The file pool counts for all device types currently defined are displayed in the following example.

```
User:   ZDFPC

System: DFPC0011I 15.00.05 15FEB DFPC  AVAILABLE FILE POOL COUNTS
                                       FILE      CORE      RATE      ORD
        SST DEVA                    189 750       880           00000001
        SDP DEVA                    138 052       550         O 00000021
        LST DEVA                     68 838         0           0000003C
        LDP DEVA                     59 160     14890        10 00000064
        4ST DEVA                     25 608         0           0000006E
            DEVB                      8 568         0           00000098
            TOT                      34 176         0
        4DP DEVA                    104 470         0         0 0000007D
            DEVB                      4 687         0         0 000000A6
            TOT                     109 157         0         0
        4D6 DEVA                        276         0         0 000000B4
            DEVB                     13 760     14900         0 000000C1
            TOT                      14 036     14900         0
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* and *TPF System Generation* for more information about pool records.

ZDGFL–Display General File Label Track

Use this command to display the data set labels for an active general file.

Requirements and Restrictions

None.

Format

```
►►──ZDGFL── ──address──────────────────────────────────────────────────►◄
```

address
is a 3- or 4-digit hexadecimal device address.

Additional Information

None.

Examples

The data set labels for the specified general file are displayed in the following example.

```
User:   ZDGFL 04E7

System: DGFL0024I 08.38.16 GENERAL FILE DATA SETS   -  -   ARE ON DEVICE 04E7, SUBSYSTEM 0003
```

Related Information

See *TPF Database Reference* for more information about general files.

ZDKAT–Display Keypoint Attributes

Use this command to display the attributes of a system keypoint record. The following information is displayed:

- Whether the keypoint is processor-shared or processor-unique
- Basic subsystem (BSS) or subsystem residency (MDBF, HPO)
- Type of residency (core resident or file resident) and addresses.

Requirements and Restrictions

None.

Format

```
►►──ZDKAT── ──KPx──────────────────────────────────────────────────────►◄
```

KP*x*
is the name of the keypoint, which can be one of the following:
A, B, C, D, E, I, M, V, 0, 1, 2, 3, 4, 5, 6, 9

Additional Information

None.

Examples

The following example displays the attributes for keypoint B.

```
User:   ZDKAT KPB

System: DKAT0001I 08.29.23 KEYPT B  IS PROC UNIQUE AND SS  RESIDENT
                           KEYPT B  FILE ADDRESS 18480017
                           KEYPT B  CORE ADDRESS 0013B000
```

Related Information

See *TPF Main Supervisor Reference* for more information about keypoints.

ZDLCK DELETE–Delete Lock Names

Use this command to delete locks from a coupling facility (CF).

Requirements and Restrictions

- Delete locks from active processors only in emergencies. No checks are performed when you enter this command and the lock is deleted. This can cause the CCCFLC program to issue system error 04627. If another processor is waiting for the lock, that processor is notified that the lock was deleted. If the lock is currently held by an entry control block (ECB) on this processor, the ECB is forced to exit, system error 0000DA is issued, and the lock is deleted from the record hold table (RHT).
- When a lock is on the CF you can only delete the lock from the processor that is holding (holder) the lock. Additionally, when the lock is on the CF and the processor is waiting for the lock, you can only delete the request for the lock from the processor that is waiting (waiter) for the lock.

Format

```
►►──ZDLCK DELete── ─┬─┬─LOCKname-lockname─┬─ ─sda─┬──────────────────────────►◄
                    │ └─LN-lockname───────┘       │
                    └─FA-fileaddr──┬──────────┬───┘
                                   └─ ─Vsync─┘
```

LOCKname-*lockname*
: deletes a specific lock, where *lockname* is the 16-digit hexadecimal lock name. You can also specify this parameter as LN.

sda
: is the 4-digit hexadecimal symbolic device address.

FA-*fileaddr*
: deletes the lock for a specific record, where *fileaddr* is an 8- or 16-digit hexadecimal file address.

  Note: To translate the file address correctly, use this parameter only for the subsystem and subsystem user (SSU) that owns the specified file address.

Vsync
: deletes the virtual file access (VFA) synchronization lock. If you do not specify this parameter, the RHT lock is deleted.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDLCK HELP
  ZDLCK ?
- Enter the ZDLCK DISPLAY command to display information about the locks that exist in the CF.
- Enter the ZRHLD DELETE command to delete locks when they are held in a control unit (CU).

Examples

In the following example, the specified lock for device 03E3 is deleted.

```
User:    ZDLCK DEL LN-0F05038a43000080 03E3

System:  CLM80027I 17.35.24 LOCK DELETION COMPLETE, CTL-4627 MAY OCCUR
```

In the following example, the VFA synchronization lock for file address F403A805 is deleted.

```
User:    ZDLCK DEL FA-F403A805 VSYNC

System:  CLM80027I 08.48.32 LOCK DELETION COMPLETE, CTL-4627 MAY OCCUR
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See “ZDLCK DISPLAY–Display Lock Names” on page 332 for more information about the ZDLCK DISPLAY command.
- See “ZRHLD DELETE–Delete Lock Names” on page 1214 for more information about the ZRHLD DELETE command.

ZDLCK DISPLAY–Display Lock Names

Use this command to display locks in a coupling facility (CF).

Requirements and Restrictions

None.

Format

```
►►──ZDLCK DISplay── ──┬─LOCKname-lockname──────┬───────────────────►◄
                      ├─FA-fileaddr────────────┤
                      └─MOD-modnumber─┬───────┬┘
                                      ├─Vsync─┤
                                      └─Rht───┘
```

LOCKname-*lockname*
: displays information about a specific lock, where *lockname* is the 16-digit hexadecimal lock name.

FA-*fileaddr*
: displays information about the locks for the specified address, where *fileaddr* is the 8- or 16-digit hexadecimal file address.

MOD-*modnumber*
: displays information about locks for the specified module, where *modnumber* is the 1- to 4-digit hexadecimal symbolic module number.

Vsync
: indicates only information about virtual file access (VFA) synchronization locks is to be displayed.

Rht
: indicates only information about the record hold table (RHT) (exclusive) locks is to be displayed.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDLCK HELP
  ZDLCK ?
- Enter the ZRHLD DISPLAY command to display information about locks that exist in control unit (CU).

Examples

The following information is displayed in the examples:

LOCKNAME
: is the address of the locked record (for example, 0000007901040000).

MCHR
: is the hexadecimal module, cylinder, head and record (MCHR) of the locked record.

SS
is the name of the subsystem that owns the record.

TYPE
is the lock type; either RHT or VFA.

STATE
is the state in which the lock is held; either EXCLUSIVE or SHARED.

SDA
is the symbolic device address (SDA) of the device where the record resides (for example, 0F20).

LOCK HOLDER LIST
is the processor identifier (ID) of one or more processors holding the record.

EXCLUSIVE LOCK WAITER LIST
is the processor ID of one or more processors waiting to get an exclusive lock on the record.

SHARED LOCK WAITER LIST
is the processor ID of one or more processors waiting to get and share the lock on the record.

FIRST WAITER
is the processor ID of the processor that gets the lock next.

All the locks for the specified file address are displayed in the following example.

```
User:    ZDLCK DISPLAY FA-F403D001

System:  CLM80009I 21.28.17 ZDLCK DISPLAY STARTS
         LOCKNAME         MCHR            SS    TYPE STATE          SDA
         0000007901040000 00470079000104  BSS   RHT  EXCLUSIVE      0EE5
         LOCK HOLDER LIST           B
         EXCLUSIVE LOCK WATIER LIST  C   _
         SHARED LOCK WAITER LIST    NONE
         FIRST WAITER                C
         LOCKNAME         MCHR            SS    TYPE  STATE         SDA
         0000007901040080 00470079000104  BSS   SYNC  EXCLUSIVE     0EE5 _
         LOCK HOLDER LIST           B
         EXCLUSIVE LOCK WAITER LIST  C
         SHARED LOCK WAITER LIST    NONE
         FIRST WAITER                C
         ZDLCK DISPLAY COMPLETE
```

The following example displays information about the specified lock.

```
User:    ZDLCK DISPLAY LOCKNAME-0000007901040000

System:  CLM80009I 08.40.54 ZDLCK DISPLAY STARTS
         LOCKNAME         MCHR            SS    TYPE STATE        SDA
         0000007901040000 00470079000104  BSS   RHT  EXCLUSIVE    0EE5
         LOCK HOLDER LIST           B
         EXCLUSIVE LOCK WAITER LIST  C   _
         SHARED LOCK WAITER LIST    NONE
         FIRST WAITER                C
         ZDLCK DISPLAY COMPLETE
```

The following example displays all the locks for a particular module. The following information is displayed in this example:

LOCKNAME
is the address of the locked record (for example, 0000007901040000).

S is the lock state, either X for exclusive or S for shared.

HOLDERS
is the lock holder list in bit format with the first processor being the high order bit in the first byte.

XWAIT
is the exclusive lock waiter list in bit format.

SWAIT
is the shared lock waiter list in bit format.

FW
is the first waiter in bit format.

```
User:    ZDLCK DISPLAY MOD-047

System:  CLM80011I 11.22.47 ZDLCK DISPLAY MOD STARTS
         LOCKNAME         S HOLDERS  XWAIT    SWAIT    FW
         0000007901040000 X 80000000 00000000 00000000 0
         0000007901080000 X 80000000 00000000 00000000 0
         0000007902040080 S 80000000 00000000 00000000 0
         ZDLCK DISPLAY MOD COMPLETE
```

Related Information

- See *TPF Database Reference* for more information about CF record lock support.
- See “ZDLCK DELETE–Delete Lock Names” on page 330 for more information about the ZDLCK DELETE command.
- See “ZRHLD DISPLAY–Display Lock Names” on page 1216 for more information about the ZRHLD DISPLAY command.

ZDLCT–Display Line Path Status

Use this command to display the current status of a line path.

Requirements and Restrictions

None.

Format

```
►►──ZDLCT── ──LN── ──line──┬──────────┬──────────────────────────────────►◄
                           ├─ ──path─┤
                           └─ ──ALL──┘
```

line
is a 2-digit hexadecimal symbolic line number.

path
is a 2-digit hexadecimal symbolic path number.

ALL
displays the status of all paths on the specified line.

Additional Information

Enter the ZALCT command to change the status of a line path.

Examples

The line path status for all the paths on line 3E is displayed in the following example.

```
User:   ZDLCT LN 3E ALL

System: 11.49.50 DLCT LN 3E
         NUM LTP IND CU SUB STATUS
        * 00  4D  D0 00 064 ACT
          01  4D  00 01 044 DOWN
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZDMAP–Display C Load Module Link Map

Use this command to display link map data about a C load module. Link map data for each C load module consists of the names and addresses of all the object files and C functions in the C load module.

Requirements and Restrictions

- You can enter this command only for C load modules.
- You must build the C load module offline in such a way that it contains link map data. See *TPF Application Programming* for more information about how to include link map data when building C load modules.

Format

```
►►──ZDMAP── ──progname──┬────────────────────────────┬──────────────────►
                        └─ ──Loadset-─┬─lsname─┬─────┘
                                      └─BASE───┘

►──┬─┬────────────────────┬─ ┬──────────────────────┬─┬────────────────►◄
   │ └─ ──Objfile-objname─┘  └─ ──Function-funcname─┘ │
   └─ ──Address-addr───────────────────────────────────┘
```

progname
: is a 4-character alphanumeric C load module name.

Loadset
: specifies the loadset that contains the C load module you want to display. If you do not specify this parameter and there is more than one version of the program active in main storage, the activation number of the entry control block (ECB) determines the version of the program that is displayed. If there are no active versions in loadsets, the base version is displayed.

  lsname
  : is the 5- to 8-character alphanumeric name of a loadset.

  BASE
  : displays the base version of the C load module.

  The LOADSET parameter is valid only after the E-type loader restart routine ends.

Objfile-*objname*
: displays link map data for one or more object files, where *objname* is the object file name. Object file names are 4 to 6 alphanumeric characters, but you can specify less than 4 characters if you use a wildcard character. Object file versions are not considered part of the object file name for purposes of matching. If you do not specify this parameter, link map data is displayed for all object files in the C load module.

Function-*funcname*
: displays link map data for one or more functions, where *funcname* is the function name. If you do not specify this parameter, link map data for functions is not displayed; only link map data for object files is displayed.

Note: Not all functions in a C load module will be included in the link map display:
- Only functions that include a standard C function prolog (functions written in C or functions written in assembler that use the TMSPC and TMSEC macros) will be included in the link map display.
- Functions that are not referenced in the C load module will not be included in the link map display.
- If you use the INLINE option to compile any of the included object files, some functions may not be included in the link map display. See the user's guide, programmer's guide, and language reference for the IBM C or C++ compiler on the System/390 platform used by your installation for more information about the INLINE option.

Address-*addr*
displays the following link map data, where *addr* is a 1- to 8-character hexadecimal address:
- The name and address of the closest function that has an address less than the specified address and the name and address of the object file that contains that function.
- The name and address of the object file that contains the specified address.
- The offset of the specified address into the listed object file.
- The name and address of the closest function in the listed object file that has an address less than the specified address (if there is one).

The C load module must be in main storage and the specified address must be in the specified C load module.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDMAP HELP
  ZDMAP ?
- The OBJFILE and FUNCTION parameters act as filters. If you specify both parameters, the link map display will include:
  – Object files that match the specified object file name and also include functions with matching function names.
  – The matching functions in those object files.
- A zero-length object file can be part of the display. Some possible reasons for a zero-length object file include:
  – The object file defines a zero-length CSECT.
  – The object file defines nothing at all.
  – The object file defines only duplicate CSECTs.
  – The object file defines only DSECTs or C structures.
  – The object file was included more than once in the build script.
- You can include an asterisk (*) in either the function name or the object file name as a wildcard character to match zero or more characters in the name. You must use the wildcard character on terminals that do not support special characters such as the underscore character (_), which may be part of a function name.
- Case is ignored when matching function names and when matching object file names.
- Addresses displayed as part of the link map display are only valid as long as the C load module remains in main storage at the same address. Lock programs in main storage by using the ZRPGM command, and avoid E-type loader activity

that could overwrite programs or remove them from main storage to ensure the addresses are valid for any length of time.

- You can display the main storage copy of a C load module contained in a loadset that was deactivated if the program allocation table (PAT) slot for the program version was not deleted. However, if the loadset is activated again, you cannot display the previous main storage copy of the C load module in the loadset because all references to that loadset use the copy of the program that was activated in main storage.
- In a loosely coupled complex, the ZDMAP command displays link map data for the program base currently used by the processor from which you enter the command. This command cannot be used to display the program link map data of any other image.
- If you enter the ZDMAP command before the E-type loader restart routine completes its processing, you cannot specify the LOADSET parameter. Therefore, you can use this command to display only the base version of a program during restart.

Examples

The following information is displayed in the examples:

COMPILED ON
: The compilation date and time of the object file in the C load module will not be displayed if the information is not available.

FUNCTION NAME
: As many as 255 characters of a function name will be displayed. The output shown in the examples is for terminals that support lowercase characters and special characters such as the underscore character (_). See *TPF System Installation Support Reference* for information about how to customize your output with a user exit.

OFFSET
: This hexadecimal value is the offset of the function into the object file.

ADDRESS
: If the address is in the range of the C load module, the name and address of the closest function that has an address less than the specified address (if there is one), the name and address of the object file that contains that function, and the offset of the specified address into the listed object file will be displayed.

The following example displays online link map information for all object files included in C load module CDM0. Only the first page is displayed. By entering ZPAGE, a second page would be displayed.

```
User:   ZDMAP CDM0

System: DMAP0001I 15.18.25 LINK MAP DATA DISPLAY
                           CDM0J1 ACTIVE IN LOADSET LINKMAP
                           C LOAD MODULE ADDRESS - 02181020
                           C LOAD MODULE SIZE    - 00003774

        CDMAINJ1 IS AN OBJECT FILE AT ADDRESS      02181098
          OBJECT FILE SIZE - 00000250
          COMPILED ON 1997/03/18 AT 10.20.26

        CDMPRSJ1 IS AN OBJECT FILE AT ADDRESS      021812F0
          OBJECT FILE SIZE - 00000520
          COMPILED ON 1997/03/18 AT 10.20.41

        CDMHLPJ1 IS AN OBJECT FILE AT ADDRESS      02181818
          OBJECT FILE SIZE - 00000118
          COMPILED ON 1997/03/18 AT 10.23.03

        CDMDSPJ1 IS AN OBJECT FILE AT ADDRESS      02181938
        MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

The following example displays the address of a specific function in the C load module. In this example, the C load module is the ISO-C library called CTAL.

```
User:   ZDMAP CTAL F-SERRC_OP

System: DMAP0001I 15.18.25 LINK MAP DATA DISPLAY
                           CTALJ1 ACTIVE IN LOADSET LINKMAP2
                           C LOAD MODULE ADDRESS - 0188D440
                           C LOAD MODULE SIZE    - 00016F90

        CSERRO40 IS AN OBJECT FILE AT ADDRESS      0189CB68
          OBJECT FILE SIZE - 000000F8
          COMPILED ON 1996/09/11 AT 14.36.16
          OFFSET    ADDRESS   FUNCTION NAME
          00000000  0189CB68  serrc_op

        END OF DISPLAY
```

The following example displays the addresses of all functions matching ZDMAP_PARSE* in C load module CDM0. Because the OBJFILE parameter is not used, the matching function and any object files that contain the matched function are displayed. For example, if a function has the same name in multiple object files in the C load module, all matching instances of the function that have a different address are displayed.

For the same result, you can also enter:

```
ZDMAP CDM0 OBJFILE-* FUNCTION-ZDMAP_PARSE*
```

```
User:   ZDMAP CDM0 FUNC-ZDMAP_PARSE*

System: DMAP0001I 15.18.25 LINK MAP DATA DISPLAY
                           CDM040 ACTIVE IN LOADSET BASE
                           C LOAD MODULE ADDRESS - 0219B020
                           C LOAD MODULE SIZE    - 00003774

        CDMPRSJ1 IS AN OBJECT FILE AT ADDRESS      0219B2F0
          OBJECT FILE SIZE - 00000520
          COMPILED ON 1997/03/18 AT 10.20.41
          OFFSET    ADDRESS   FUNCTION NAME
          00000000  0219B2F0  zdmap_parse

        CDMER1J1 IS AN OBJECT FILE AT ADDRESS      0219CC18
          OBJECT FILE SIZE - 00000140
          COMPILED ON 1997/03/18 AT 10.20.53
          OFFSET    ADDRESS   FUNCTION NAME
          00000000  0219CC18  zdmap_parse_error_handler

        END OF DISPLAY
```

In the following example, the asterisk (*) is used as a wildcard character to request all functions beginning with *process* in C load module CDM2 in loadset LINKMAP in object file CDMPRC.

The following are some examples with specific letters:
- *process** yields all functions (and the object files in which they are contained) that begin with *process*.
- **process* yields all functions (and the object files in which they are contained) that end with *process*.
- **process** yields all functions (and the object files in which they are contained) that contain character string *process*.
- *process*it** yields all functions (and the object files in which they are contained) that begin with *process* and contain character string *it*.

```
User:   ZDMAP CDM2 L-LINKMAP O-CDMPRC F-PROCESS*

System: DMAP0001I 15.18.25 LINK MAP DATA DISPLAY
                           CDM2J1 ACTIVE IN LOADSET LINKMAP
                           C LOAD MODULE ADDRESS - 021C2020
                           C LOAD MODULE SIZE    - 00001F3C

        CDMPRCJ1 IS AN OBJECT FILE AT ADDRESS      021C2098
          OBJECT FILE SIZE - 00001C18
          COMPILED ON 1997/03/18 AT 10.24.13
          OFFSET    ADDRESS   FUNCTION NAME
          000002A0  021C2338  process_Address
          00000618  021C26B0  process_one_object_file
          00000FE8  021C3080  process_one_function
          00001238  021C32D0  process_Objfile_and_Function
          00001848  021C38E0  process_return_storage

        END OF DISPLAY
```

The following example displays the closest object file and function that has an address less than the specified address. The offset of the specified address into the object file is also displayed.

```
User:   ZDMAP CDM3 A-21C7100

System: DMAP0001I 15.18.25 LINK MAP DATA DISPLAY
                   CDM340 ACTIVE IN LOADSET BASE
                   C LOAD MODULE ADDRESS - 021C7020
                   C LOAD MODULE SIZE    - 00000268

        CDMRELJ1 IS AN OBJECT FILE AT ADDRESS      021C7098
          OBJECT FILE SIZE - 00000160
          COMPILED ON 1997/03/18 AT 10.24.01
          OFFSET    ADDRESS   FUNCTION NAME
          00000000  021C7098  tpf_cdm3_release_locks

        021C7100 IS AT OFFSET 000068 INTO OBJECT FILE CDMRELJ1

        END OF DISPLAY
```

Related Information

- See *TPF System Installation Support Reference* for more information about customizing your displayed output of C load module link map data with a user exit.
- See *TPF Program Development Support Reference* for information about how C load module link map data is formatted in a dump.

ZDMFS–Display Module File Status

Use this command to display the module file status table (MFST) information for one or more direct access storage devices (DASDs) assigned to the basic subsystem (BSS) or to a subsystem in a multiple database function (MDBF) environment.

Note: In some places, a DASD is also referred to as a module.

Requirements and Restrictions

None.

Format

```
►►──ZDMFS──┬─────────────────────────────┬──────────────────────────────►◄
           │                ┌─1───────┐  │
           ├─ ──module── ───┼─────────┼──┤
           │                ├─number──┤  │
           │                └─Online──┘  │
           └─ ──Online───────────────────┘
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time DASD or general data sets, or a pseudo module number for general file DASD.

  Note: If you do not specify a module number, information about the lowest module number is displayed.

number
: is the number of DASD to be displayed from 0–18.

Online
: displays information about the online DASD.

Additional Information

None.

Examples

The following information is displayed in the example:

MOD
: is the symbolic or pseudo module number.

TYPE
: is the DASD model number and one of the following indicators:

  ECKD
  : Module is running in extended count key data mode.

  C/D
  : Module is running in count key data mode on a caching control unit.

C/E
Module is running in extended count key data mode on a caching control unit.

RCS
Module is running in extended count key data mode on a full function record caching control unit.

SDA
is the symbolic device address.

USE
is the way the module is being used, where:

GEN
Module is a pseudo general file.

RLT
Module is a real-time file.

GDS
Module is a general data set.

NMT
Module is not mounted.

DUP
is the symbolic module number of the duplicate module, if one exists.

STAT
is the module status. The possible conditions are:

ON
Module is online.

OFF
Module is offline.

COPY
Module is being copied ***from*** or is being copied ***to*** by the ZMCPY function.

VSN
is the volume serial number.

LOCK
is the lock status, where:

CF
Module is running on a coupling facility.

CFLF
Module is running on a concurrency filter lock facility.

LLF
Module is running on a limited lock facility.

NONE
Module currently has no locking.

COR
is the correctable error count (in decimal).

UNC
is the uncorrectable error count (in decimal).

PREFIX
indicates one of the following:

YES
Module is running on a control unit that supports the prefixing architecture.

NO
Module is running on a control unit that does not support the prefixing architecture.

The following example displays the MFST information for six modules (00A – 00F).

```
User:   ZDMFS A 6

System: DMFS0004I 08.41.39
              MOD TYPE     SDA  USE DUP STAT  VSN   LOCK COR   UNC   PREFIX
              00A          N/A  GEN N/A  OFF
              00B          N/A  GEN N/A  OFF
              00C          N/A  GEN N/A  OFF
              00D 3380     04E7 GEN N/A   ON BP0999 NONE 00000 00000   NO
              00E 3380     04E8 RLT 0582  ON BP8372 LLF  00000 00000   NO
              00F 3380     04EC RLT N/A   ON BP0456 CF   00000 00000   NO
        DISPLAY COMPLETE
```

Related Information

None.

ZDMOD–Display DASD Information

Use this command to display the following information about a specified direct access storage device (DASD):

- The main storage address for the module file status table (MFST)
- The symbolic device number
- The queue length
- The main storage address for the control unit status table.

Requirements and Restrictions

None.

Format

```
►►──ZDMOD── ──module──────────────────────────────────────────────►◄
```

module
is the 3-digit hexadecimal symbolic module number.

Additional Information

None.

Examples

The information for the 047 DASD are displayed in the following example, where:

FST0
is the address of section 0 of the module file status table (MFST).

FST1
is the address of section 1 of the MFST.

DNBR
is the symbolic device number.

QCNT
is the queue length.

ACST
is the address of the control unit status table.

```
User:   ZDMOD 047

System: CFD20004I MODULE 047
        FST0    01EB9000
        FST1    01EB9400
        DNBR    00000EE5
        QCNT    00000000
        ACST    01EDBDC0
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZDPAT–Display Program Allocation Table

Use this command to display the values assigned to the allocation parameters of a program. The following information is displayed:
- Program version code
- Program type
- Program class
- File address of the program
- Addressing mode in which the program was entered (24-bit or 31-bit)
- Macros that the program is authorized to use.

Requirements and Restrictions

None.

Format

```
                          ┌ ──Copy-F──────┐
►►──ZDPAT── ──progname────┼───────────────┼──────────────────────────────────►◄
                          └ ──Copy-──┬C─┬─┘
                                     └B─┘
```

progname
: is a 4-character alphanumeric program name.

Copy
: specifies which copy of the PAT to display:

  F displays the file copy.

  C displays the core copy.

  B displays both the core and the file copy.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDPAT HELP
  ZDPAT ?
- If the program is a transfer vector, information about the parent program is displayed rather than information about the transfer vector.
- If you display the core copy of the program allocation table (PAT), the values of the allocation parameters of the allocated version of the program and all versions of the program in core that were activated using the E-type loader are displayed.
- Enter the ZAPAT command to change the allocation parameters for programs in the PAT.

Examples

In the following example, the core copy of the PAT entry for the CVAA program is displayed. The active image is TPF01.

The information displayed in the example is for the core copy of the USOK program. The linkage type of the USOK program is displayed as an abbreviation, DLM. This means the USOK program uses the dynamic link module (DLM) linkage protocol. The linkage types are abbreviated according to the following table.

Abbreviation	Type	Description
BAL	Assembler	Basic assembler language
DLM	Dynamic Load Module	ISO-C dynamic load module, including dynamic link library (DLL) application
DLL	Dynamic Link Library	Dynamic link library module
LIB	Library	ISO-C language library module
TAR	TARGET(TPF)	TARGET(TPF) C language

```
User:   ZDPAT USOK C-C

System: DPAT0102I 16.36.13
        BEGIN DISPLAY OF CORE COPY

         PROGRAM         USOK
         VERSION         40
         BASE PAT SLOT   062E3F10
         TYPE            CORE RESIDENT PRELOAD NODBUG
         LINKAGE TYPE    DLM
         CLASS           SHARED

         FILE ADDRESS    002C3519
         ADDRESSING MODE 31BIT
         AUTHORIZATION   NONE

         C DEBUGGER HOOK NONE
         ADATA FILE ADDR ADATA NOT LOADED
        DISPLAY OF PAT SLOTS FOR USOK

        VV LOADSET  ACT NUM STAT  FILE ADDR TYPE LINK HOOKS PAT ADDR ADATA FA
        -- -------- ------- ----- --------- ---- ---- ----- -------- --------
        FA NATIVEIP      1  ACT   004403D5   CR  DLM   ON   03CBB018 N/A
        40 BASE          0  ACT   002C3519   CR  DLM  N/A   062E3F10 N/A
        END OF DISPLAY
```

The PAT slots of the two active versions of the USOK program show that version 40 is in the BASE loadset and uses the DLM linkage protocol. Version FA is in the NATIVEIP loadset and uses the DLM linkage protocol.

In the following example, a display of both the file copy and the core copy of the PAT entry for the USOK program is requested.

```
User:   ZDPAT USOK COPY-B

System: DPAT0101I 15.48.38 BEGIN DISPLAY OF FILE COPY FOR IMAGE PUT13A

         PROGRAM        USOK

         VERSION        40
         BASE PAT SLOT   062E4F10
         TYPE           CORE RESIDENT PRELOAD NODBUG
         LINKAGE TYPE    DLM
         CLASS          SHARED

         FILE ADDRESS    002C3519
         ADDRESSING MODE 31BIT
         AUTHORIZATION   NONE

         C DEBUGGER HOOK NONE
         ADATA FILE ADDR ADATA NOT LOADED

        BEGIN DISPLAY OF CORE COPY

         PROGRAM        USOK

         VERSION        40
         BASE PAT SLOT   062E4F10
         TYPE           CORE RESIDENT PRELOAD NODBUG
         LINKAGE TYPE    DLM
         CLASS          SHARED

         FILE ADDRESS    002C3519
         ADDRESSING MODE 31BIT
         AUTHORIZATION   NONE

         C DEBUGGER HOOK NONE
         ADATA FILE ADDR ADATA NOT LOADED
        DISPLAY OF PAT SLOTS FOR USOK

        VV LOADSET  ACT NUM STAT  FILE ADDR TYPE LINK HOOKS PAT ADDR ADATA FA
        -- -------- ------- ----- --------- ---- ---- ----- -------- -------- _
        FA NATIVEIP      2  ACT   004403D5   CR  DLM   ON   03D35088 N/A
        40 BASE          0  ACT   002C3519   CR  DLM  N/A   062E4F10 N/A
        END OF DISPLAY
```

Related Information

- See *TPF Main Supervisor Reference* and *TPF System Installation Support Reference* for more information about the program allocation table (PAT).
- See *TPF Program Development Support Reference* for more information about C language register environments in various linkage protocols.
- See *TPF Application Programming* for information about linkage types.

ZDPGM–Display Program

Use this command to display a specified number of bytes from a program beginning at a relative starting address. You can display real-time programs (both file resident and core resident), control programs, and core image restart area programs (FCTB, ACPL, ICDF, IPLB, SIGT, RIAT, and IPAT).

Requirements and Restrictions

- The LOADSET parameter and the *version* variable are valid only after the E-type loader restart routine ends.
- This command cannot be used to display the following programs:
  - The core copy of private programs cannot be displayed.
  - The core copy of an I-stream unique program. To display the core copy, use the ZRPGM command to lock the program in core and then use the ZDCOR command to display the program.
  - The core copy of ACPL, IPLA, and IPLB, because the core copy is only in storage at restart time. The core copy cannot be displayed.
- You cannot enter this command for the file copies of C load modules.

Format

```
►►──ZDPGM── ──progname──┬──────────┬──┬───────────────────────┬──────────────►
                        └─.version─┘  │    ┌─0───┐  ┌─.20────┐│
                                      └────┼─────┼──┼────────┼┘
                                           └─rsa─┘  └─.bytes─┘

                                                  ┌──Copy-F──────┐
►──┬─────────┬──┬──────────────────────────┬──┼──────────────┼──────────►
   └──Instr──┘  └──Loadset-──┬─lsname─┬────┘  └──Copy-──┬─C─┬─┘
                             └─BASE───┘                 └─B─┘

   ┌──Type-R──────┐
►──┼──────────────┼─────────────────────────────────────────────────────►◄
   └──Type-──┬─P─┬┘
             └─D─┘
```

progname
: is a 4- to 6-character alphanumeric program name.

version
: is a 2-character alphanumeric program version code. This parameter can be specified only for real-time programs.

rsa
: is the 1- to 6-digit hexadecimal relative starting address (offset) in the program listing.

bytes
: is the hexadecimal number of bytes to display from X'01'–X'FFF'.

  Note: The maximum number of bytes for CIMR and IPL components is X'FEF'.

Instr
displays the data in the disassembled format rather than the hexadecimal format.

Note: You can display only as many as X'400' bytes at a time in disassembled format.

Loadset
specifies the loadset that contains the program you want to display.

lsname
is the 5- to 8-character alphanumeric name of a loadset.

BASE
indicates that you want to display the base version of the program. You can specify this option for the LOADSET parameter only for real-time programs.

Copy
specifies which copy of the program to display:

F displays the file copy.

C displays the core (main storage) copy.

B displays both the file and core copies.

Type
specifies which copy on file to display.

R displays either the prime or the duplicate copy of the program.

P displays the prime copy of the program.

D displays the duplicate copy of the program.

Note: This parameter is ignored if you specify the C option for the COPY parameter.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDPGM HELP
  ZDPGM ?
- If you specify a relative starting address that does not begin on a fullword boundary, the program adjusts that address to the next lower fullword boundary. However, if you specify the INSTR parameter, the program adjusts the address to the next lower halfword address.
- If you do not specify a program version code or a loadset name and there is more than one version of the program active in core, the activation number of the ECB determines the version of the program that is displayed.
- You can display the file copy of a real-time program in a loadset even if the loadset is not active.
- You can display the core copy of a real-time program contained in a loadset that was deactivated if the PAT slot for the program version was not deleted. However, if the loadset is activated again, you cannot display the old core copy of the real-time program in the loadset because all references to that loadset use the copy of the program that was activated in core.
- The ZDPGM command displays program records in the image that is currently used by the processor. It cannot be used to display the real-time program records of any other image. If the CIMR component on the current image

logically references another image, the logical copy is displayed. In this case, the response message specifies the actual image that was displayed.
- If you enter the ZDPGM command before the E-type loader restart routine completes, you cannot specify the Loadset parameter or the *version* variable. Therefore, you can use this command to display only the base version of a program.
- Enter the ZAPGM command to change data in a program.

Examples

The following example displays the first 20 bytes of the CVAP program in EBCDIC format.

```
User:   ZDPGM CVAP

System: DPGM0010I 14.30.56 BEGIN DISPLAY OF FILE COPY FOR
                   CVAP.40 ACTIVE IN LOADSET BASE
         00000000- 00FF0E4E C3E5C1D7 95019115 47808014 ....CVAP n.j.....
         00000010- 0A320010 9501911D 47808020 0A320018 ....n.j. ........ _
         END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays 30 bytes of the prime file copy of the CVAP program beginning at a relative starting address of X'28A'.

```
User:   ZDPGM CVAP 28A.30 TYPE-P

System: DPGM0010I 14.31.23 BEGIN DISPLAY OF FILE COPY FOR
                  CVAP.40 ACTIVE IN LOADSET BASE
         00000288- 401047F0 827CD505 40209390 4780831E  ..0b@N.  .l...c.
         00000298- 41404010 4630827C 12004780 82A81840 .  ...b@ ....by.  _
         000002A8- 0A2C1851 185ED203 50008E04 D7015010 .....;K. &;..P.&;
         000002B8- 5010                                &;
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays the CVAP program in disassembled format.

```
User:   ZDPGM CVAP 28A.20 INST

System: DPGM0010I 15.28.54 BEGIN DISPLAY OF FILE COPY FOR
                   CVAP.40 ACTIVE IN LOADSET BASE
        0000028A  47F0 827C       B      636(,R8)
        0000028E  D505 4020 9390  CLC    32(6,R4),912(R9)
        00000294  4780 831E       BC     8,798(,R8)
        00000298  4140 4010       LA     R4,16(,R4)
        0000029C  4630 827C       BCT    R3,636(,R8)
        000002A0  1200            LTR    R0,R0
        000002A2  4780 82A8       BC     8,680(,R8)
        000002A6  1840            LR     R4,R0
        000002A8  0A2C            GETCC
        END OF DISPLAY
```

Related Information

See *TPF Main Supervisor Reference* for more information about displaying program data.

ZDPLT–Display Program Linkage Type

Use this command to do the following:

- Display program names from the program allocation table (PAT) and the extra PAT slots with the specified linkage type.
- Display the date and time when the PAT was created.

Requirements and Restrictions

None.

Format

```
►►──ZDPLT── ──┬─┬─BAL─────┬──┬──────────────────┬──┬───────────────────────►◄
              │ ├─DLL─────┤  └── ──Name-progname─┘  │
              │ ├─DLM─────┤                         │
              │ ├─LIBrary─┤                         │
              │ └─TARget──┘                         │
              └─┬─Ipat──────────┬───────────────────┘
                └─Name-progname─┘
```

BAL
: displays basic assembler language programs.

DLL
: displays C language dynamic link libraries.

DLM
: displays C language dynamic load modules.

LIBrary
: displays program libraries.

TARget
: displays TARGET(TPF) programs.

Ipat
: displays the date and time when the PAT was created.

Name-*progname*
: specifies the name of the program for which to search in the PAT and extra PAT slots, where *progname* is an alphanumeric program name or search pattern. A wildcard character (*) can be used in a valid search pattern. The pattern length cannot exceed 4 characters and can only be less than 4 characters when the last character is a wildcard character (*). If you do not specify this parameter, all programs in the PAT with the specified linkage type will be displayed. If you specify this parameter alone, all programs in the PAT which match the search pattern will be displayed regardless of linkage type.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDPLT HELP
  ZDPLT ?

- If no matches are found using the search criteria entered, NO MATCHES FOUND is displayed instead.
- Normally, program names are sorted and listed in the display alphabetically by the program name. However, if you enter the ZDPLT command during restart, the program names are not sorted. In addition, there may be times that program names cannot be sorted because of system resource constraints. If this occurs, and you want a sorted display, enter the ZDPLT command again at a later time.

Examples

The following example shows the use of the wildcard character in the NAME parameter.

```
User:   ZDPLT BAL NAME-CCD*

System: DPLT0002I 11.07.31 DISPLAY ASSEMBLER PGMS LOADED OR ALLOCATED
                7 PGMS FOUND
          CCDA40-BAL  CCDB40-BAL  CCDC40-BAL  CCDD40-BAL  CCDE40-BAL
          CCDT40-BAL  CCD1  -BAL
        END OF ASSEMBLER DISPLAY
```

The following example displays all programs with the DLL linkage type.

```
User:   ZDPLT DLL

System: DPLT0003I 10.46.41 DISPLAY ISO-C DLLS LOADED
                5 DLLS FOUND
          QPNB41-DLL  QPN541-DLL  QPN641-DLL  QPN741-DLL  QPN841-DLL
        END OF ISO-C DLL DISPLAY
```

The following example shows what happens when the specified program is not found in the PAT or extra PAT slots.

```
User:   ZDPLT LIBRARY NAME-QZZ9

System: DPLT0005I 15.46.56 DISPLAY ISO-C LIBRARIES LOADED
                NO MATCHES FOUND
        END OF ISO-C LIBRARY DISPLAY
```

The following example displays the date and time when the PAT was created.

```
User:   ZDPLT IPAT

System: DPLT0008I 15.38.24 DISPLAY IPAT INFORMATION
                IPAT CREATION TIME MON FEB  2 18:13:09 1998
        END OF IPAT DISPLAY
```

The following example displays all programs listed in the PAT and extra PAT slots that have a name beginning with CVR regardless of linkage type.

```
User:   ZDPLT NAME-CVR*

System: DPLT0006I 11.10.13 DISPLAY PGM NAMES LOADED OR ALLOCATED
                8 PGMS FOUND
          CVRA40-BAL  CVRB  -BAL  CVRI40-DLM  CVRM40-BAL  CVRN40-BAL
          CVRO  -BAL  CVRP  -BAL  CVRQ40-BAL
        END OF PROGRAM NAME DISPLAY
```

Related Information

- See *TPF Main Supervisor Reference* and *TPF System Installation Support Reference* for more information about the program allocation table (PAT)
- See *TPF Application Programming* for information about linkage types.

ZDRCT–Display the RCAT

Use this command to display information about an application.

Requirements and Restrictions

None.

Format

```
►►──ZDRCT── ──┬─ALL──────────────┬──┬────────────┬──────────────────────►◄
              ├─APPL-name────────┤  ├─,ACTive────┤
              ├─CPU-id───────────┤  ├─,SHUTdown──┤
              ├─SSN-subsystem────┤  └─,STOPped───┘
              └─SSUN-username────┘
```

ALL
displays information about all the applications defined in the routing control application table (RCAT).

APPL-*name*
displays information about the specified application, where *name* is the 4-character alphanumeric name of an application.

CPU-*id*
displays information about all of the applications defined for the specified processor, where *id* is the 1-character alphanumeric CPU ID of a processor.

SSN-*subsystem*
displays information about the applications defined for the specified subsystem, where *subsystem* is the 1- to 4-character alphanumeric name of a subsystem.

SSUN-*username*
displays information about the applications defined for the specified subsystem user, where *username* is the 1- to 4-character alphanumeric name of a subsystem user.

ACTive
displays only the active applications.

SHUTdown
displays only the applications that are shut down.

STOPped
displays only the applications that are stopped.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZDRCT HELP
ZDRCT ?

Examples

The following information is displayed in the examples:

APPL
    is the name of the application.

CPU
    is the CPU ID of the application.

SS
    is the subsystem name of the application.

SSU
    is the subsystem user name of the application.

STATUS
    is the status of the application, which can be ACTIVE, STOPPED, or SHUTD.

NCB
    is the number of the NCB slot in the NCB directory record entry that a dynamic LU will use when it logs on to the application. The NCB slot contains the address of the NCB record.

    Note: The NCB slot for the application is defined using the MSGRTA macro.

Information about the CXFT application is displayed in the following example.

```
User:   ZDRCT APPL-CXFT

System: DRCT0001I 12.14.24
        APPL CPU SS   SSU  STATUS   NCB
        CXFT B   BSS  HPN  STOPPED   05
        END OF RCAT DISPLAY
```

Information about all of the active applications in the basic subsystem (BSS) is displayed in the following example.

```
User:   ZDRCT SSN-BSS,ACTIVE

System: DRCT0001I 12.15.51
        APPL CPU SS   SSU  STATUS   NCB
        AAAA A   BSS  HPN  ACTIVE    03
        CLGA A   BSS  HPN  ACTIVE    02
        CLGB B   BSS  HPN  ACTIVE    05
        CLGC C   BSS  HPN  ACTIVE    01
        CLGD D   BSS  HPN  ACTIVE    02
        CLGE E   BSS  HPN  ACTIVE    01
        CLGZ Z   BSS  HPN  ACTIVE    01
        CLG0 0   BSS  HPN  ACTIVE    00
        LOGI B   BSS  HPN  ACTIVE    00
        SMPA A   BSS  HPN  ACTIVE    03
        SMPB B   BSS  HPN  ACTIVE    03
        SMPC C   BSS  HPN  ACTIVE    01
        SMPD D   BSS  HPN  ACTIVE    01
        SMPE E   BSS  HPN  ACTIVE    00
        SMPZ Z   BSS  HPN  ACTIVE    00
        SMP0 0   BSS  HPN  ACTIVE    00
        END OF RCAT DISPLAY
```

Related Information

- See *TPF Data Communications Services Reference* for more information about the message routing package.
- See *TPF System Generation* for more information about the MSGRTA macro.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB directory records.

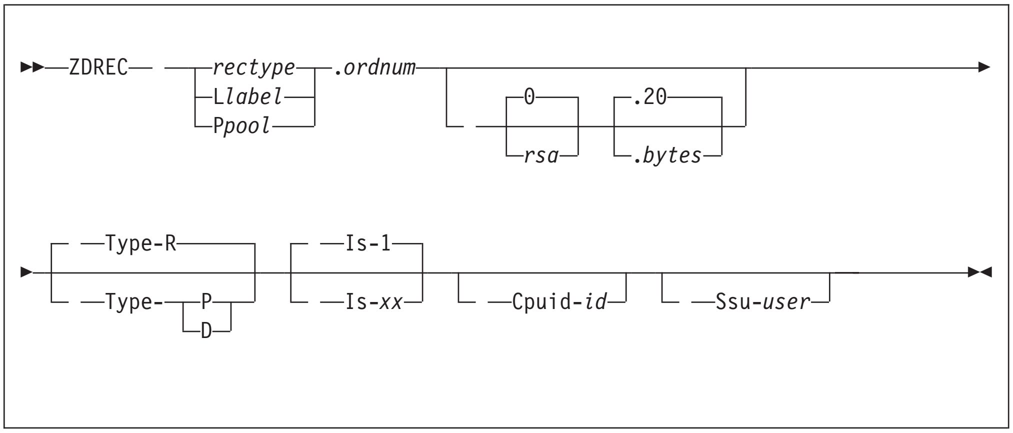
ZDREC–Display a Fixed File or Pool Record

Use this command to display a specified number of bytes from a record beginning at a relative starting address. The file address of the record is also displayed.

Requirements and Restrictions

None.

Format



rectype
: is a 1- to 4-digit hexadecimal record type (for FACE).

L*label*
: is a 1- to 8-character SYSEQ label (for FACS).

Note: The number sign (#) that precedes the SYSEQ label is optional unless you need to distinguish between 2 labels; for example, #RECA and RECA.

P*pool*
: is a pool record, where *pool* is one of the following pool record types:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

ordnum
is a 1- to 16-digit hexadecimal ordinal number.

rsa
is a 1- to 3-digit hexadecimal relative starting address (offset) in the record.

bytes
is the hexadecimal number of bytes to display from X'01'–X'FFF'.

Type
specifies which copy of the record to display:

R displays either the prime or duplicate copy of the record.

P displays the prime copy of the record.

D displays the duplicate copy of the record.

Is-*xx*
specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Cpuid-*id*
is the 1-character alphanumeric CPU ID of the processor.

Ssu-*user*
is a subsystem user.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDREC HELP
  ZDREC ?
- If the address you specify is not on a fullword boundary, the display begins at the next lower fullword.
- Enter the ZAREC command to change the data in a record.

Examples

The following example displays 40 bytes from a fixed file record.

```
User:   ZDREC 87.0 .40 TYPE-P

System: DREC0011I 15.28.54 DISPLAY OF FILE ADDRESS 00000000F403D001
         00000000- C1D60080 C3C7E3F2 00000514 00000515 AO..CGT2 ........
         00000010- 00030FE8 0000432C 00004EB4 000023D0 ...Y.... ........
         00000020- 00002538 00000000 00000000 00000000 ........ ........
         00000030- 00000000 00000000 00002A24 00003564 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays a record using the #UATRI SYSEQ label.

```
User:   ZDREC LUATRI.9 108.30

System: DREC0011I 15.28.54 DISPLAY OF FILE ADDRESS 00000000F400D025
         00000108- FE1442C2 D3C1E7A8 18210000 00000000 ...BLAXy ........
         00000118- 00000000 FE1443C2 D3C1E76A 18A10000 .......B LAX..s..
         00000128- 00000000 00000000 FE1444C2 D3C1E7A8 ........ ...BLAXy
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays data from a pool record.

```
User:   ZDREC PLST.40

System: DREC0011I 15.28.54 DISPLAY OF FILE ADDRESS 0000000000040101
         00000000- 00000000 00000000 00000000 00000000 ........ ........
         00000010- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Concepts and Structures* for more information about fixed file records and pool records.

ZDSER–Display System Error Options

Use this command to display the system error options that are set for system error dumps (SERRC) or the SNAP options that are set for SNAP dumps (SNAPC). These options determine what course of action to follow if an error occurs.

Requirements and Restrictions

None.

Format

```
►►──ZDSER──┬────────┬──────────────────────────────────────────────►◄
           └──SNAP──┘
```

SNAP
displays the SNAP dump options. If you do not specify the SNAP parameter, the system error options are displayed.

Additional Information

- Use the ZASER command to change the system error options for SERRC dumps and the SNAP options for SNAPC dumps.
- Table 4 and Table 5 on page 361 list the status for the different system error and SNAP dump options and provide the ZASER command you can enter to set each option.

  The *status* variable indicates the possible status for the option; for example, ON or OFF.

Table 4. System Error Options for SERRC Dumps

Text in ZDSER Display	**Possible** *status*	**Corresponding ZASER Command**
`DATA TO DUMP DEVICE –` *status*	OFF	ZASER DUMPOFF
	ON	ZASER DUMPON
`DUMP HEAP FOR THREAD DUMPS –` *status*	OFF	ZASER THEAPOFF
	ON	ZASER THEAPON
`MSG TO CONSOLE –` *status*	OFF	ZASER CONSOFF
	DMP ONLY	ZASER CONSDMP
	ON	ZASER CONSON
`MSG TO TAPE –` *status*	OFF	ZASER NOLOG
	ON	ZASER LOG
status `OPR DUMP`	SHORT	ZASER SHORT
	MED	ZASER MEDIUM
	LONG	ZASER LONG
status `DUPLICATE ECB ERRORS`	SUPPRESS	ZASER NODUPL
	ALLOW	ZASER DUPL

Table 4. System Error Options for SERRC Dumps (continued)

Text in ZDSER Display	Possible *status*	Corresponding ZASER Command
LOG SYSTEM DUMPS TO *status*	TAPE	ZASER TAPE
	PRT	ZASER PRT
LOG ECB ASSOCIATED DUMPS TO *status*	TAPE	ZASER TAPE
	PRT	ZASER PRT ECB CNTRLD
DUMP TAPE – *status*	RTA RTL	
DATA TO DUMP DATA USER EXIT – *status*	OFF	ZASER NODATAX
	ON	ZASER DATAX

Table 5. SNAP Options for SNAPC Dumps

Text in ZDSER Display	Possible *status*	Corresponding ZASER Command
SNAP DATA TO DUMP DEVICE – *status*	OFF	ZASER SNAP DUMPOFF
	MSG	ZASER SNAP DUMPMSG
	ON	ZASER SNAP DUMPON
SNAP DATA TO CONSOLE – *status*	OFF	ZASER SNAP CONSOFF
	MSG	ZASER SNAP CONSMSG
	ALL	ZASER SNAP CONSALL
status DUPLICATE SNAP ERRORS	SUPPRESS	ZASER SNAP NODUPL
	ALLOW	ZASER SNAP DUPL
LOG SNAP DUMPS TO *status*	TAPE	ZASER SNAP TAPE
	PRT	ZASER SNAP PRT
DUMP TAPE – *status*	RTA RTL	
DATA TO DUMP DATA USER EXIT – *status*	OFF ON	

- The value for the DUMP TAPE options is defined in keypoint record A (CTKA) during SIPGEN and cannot be changed using the ZASER command.
- There are no multiple database function (MDBF) subsystem-unique options.

Examples

The following example displays the system error dump options.

```
User:   ZDSER

System: DSER0001I 14.10.39
        DATA TO DUMP DEVICE - ON
        MSG TO CONSOLE - ON
        MSG TO TAPE - OFF
        MED. OPR DUMP
        SUPPRESS DUPLICATE ECB ERRORS
        LOG SYSTEM DUMPS TO TAPE
        LOG ECB ASSOCIATED DUMPS TO TAPE
        DUMP TAPE - RTA
        DATA TO DUMP DATA USER EXIT - OFF
        DUMP HEAP FOR THREAD DUMPS - OFF
```

The following example displays the SNAPC dump options.

```
User:   ZDSER SNAP

System: DSER0002I 14.17.47
        SNAP DATA TO DUMP DEVICE - ON
        SNAP DATA TO CONSOLE - ALL
        SUPPRESS DUPLICATE SNAP ERRORS
        LOG SNAP DUMPS TO TAPE
        DUMP TAPE - RTA
        DATA TO DUMP DATA USER EXIT - OFF
```

Related Information

See *TPF General Macros* for more information about system error dumps (SERRC) and SNAP dumps (SNAPC).

ZDSID–Display System ID

Use this command to display the system ID. The system ID consists of 3 components:

- Enterprise name, which is the name of your operations center or enterprise
- TPF complex name, which is the name of your loosely coupled complex or stand-alone TPF processor
- CPU ID, which is the CPU identifier.

Requirements and Restrictions

None.

Format

```
►►──ZDSID──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example displays the system ID.

```
User:   ZDSID

System: DSID0001I 14.21.14 THE SYSTEM ID IS  DANBURY.TPFNET   B
        ON PROCESSOR FF020410 MODEL 3090 LOW CPU ADDRESS 0000
```

Related Information

See the CONFIG macro in *TPF System Generation* for more information about system IDs.

ZDSMG DEFINE–Define a Data Definition

Use this command to define a data definition name to the TPF system. You can define a data definition name for a tape device, general file data set, virtual reader, or another user-defined medium.

Requirements and Restrictions

- The data definition name is defined only for the current processor.
- You cannot define a data definition name for a general data set with this command. Use the ZDSMG MT command to define a data definition name when you mount the general data set.

Format

```
►►──ZDSMG DEFine── ──ddname── ──┬─Tape-name──────────────┬──────────────────────►◄
                                ├─Gfds-number────────────┤
                                ├─Vrdr-──┬─spoolid─┬─────┤
                                │        ├─ALL─────┤     │
                                │        └─FIFO────┘     │
                                └─User-──┬─variable───┬──┘
                                         ├─'variable'─┤
                                         └─$variable$─┘
```

ddname
: is a 1- to 16-character alphanumeric name that you want to assign to the data set.

Tape-*name*
: is a 3-character alphanumeric tape name.

GFDS-*number*
: is a general file data set number from 0–59.

Vrdr-*spoolid*
: is the spool ID of a file in the virtual reader.

ALL
: specifies all files in the virtual reader.

FIFO
: specifies the first file in the virtual reader.

User-*variable*
: is a 1- to 44-character alphanumeric name that identifies the user-defined medium.

  If the name contains imbedded blanks or other special characters, begin and end the name with the single quotation mark (') or the dollar sign ($) character. The TPF system removes these characters from the beginning and ending name.

Additional Information

None.

Examples

The TESTDSN data definition name is defined for the OLD tape device in the following example.

```
User:   ZDSMG DEF TESTDSN TAPE-OLD

System: DSMG0001I DDNAME TESTDSN       DEFINED
```

The TESTDSN data definition name is defined for general file data set number 3 in the following example.

```
User:   ZDSMG DEF TESTDSN GFDS-3

System: DSMG0001I DDNAME TESTDSN       DEFINED
```

In the following example, the TESTDSN data definition name is defined for the virtual reader spool file that has a spool identifier of 1823.

```
User:   ZDSMG DEF TESTDSN VRDR-1823

System: DSMG0001I DDNAME TESTDSN       DEFINED
```

In the following example, the TESTDSN data definition name is defined for a user-defined medium that is identified to the TPF system as MIPC ACCESS TO SYSTEM C. Notice that this string begins and ends with the single quotation mark (').

```
User:   ZDSMG DEF TESTDSN USER-'MIPC ACCESS TO SYSTEM C'

System: DSMG0001I DDNAME TESTDSN       DEFINED
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG DISPLAY–Display Data Set

Use this command to display information about a data definition or a data set.

Requirements and Restrictions

None.

Format

```
                          ┌─ ──────────────┐
                          ▼                │
►►──ZDSMG DISplay── ──┬──┬─DDName-ddname─┬┴──┬──────────────────────────────►◄
                      │  └─DSName-dsname─┘   │
                      └─ALL──────────────────┘
```

DDName-*ddname*
: displays information about a specific data definition, where *ddname* is a 1- to 16-character alphanumeric data definition name.

DSName-*dsname*
: displays information about a specific data set, where *dsname* is a 1- to 44-character alphanumeric data set name.

ALL
: displays information about all the mounted data definitions and data sets.

Additional Information

If you specify both a data definition name and a data set name, this function verifies that the data set name you specified matches the data definition name in the system data set control block (DSCB).

Examples

Information about the TESTDSN data definition is displayed in the following example.

```
User:   ZDSMG DISP DDN-TESTDSN

System: DSMG0108I 08.44.16 START OF ZDSMG DISPLAY OUTPUT
        DDN  TESTDSN          TYPE TAPE
        PARAMETER OLD
        DSMG0010I 08.44.16 DISPLAY COMPLETE
```

Information about the TPF.TEST.G0002V00 data set is displayed in the following example.

```
User:   ZDSMG DISP DSN-TPF.TEST.G0002V00

System: DSMG0108I START OF ZDSMG DISPLAY OUTPUT
        DDN  GNFLBSS          NUMBER VOLUMES MOUNTED  1
        DSN  TPF.TEST.G0002V00
        DSMG0009I VOLUME NUMBER  1 VOLUME LABEL BP0999
        DSMG0010I DISPLAY COMPLETE
```

Information about all of the data definitions and mounted data sets is displayed in the following example.

```
User:   ZDSMG DISP ALL

System: DSMG0108I START OF ZDSMG DISPLAY OUTPUT
        DDN  TESTDSN       TYPE TAPE
        PARAMETER OLD
        DDN  TESTDS        TYPE USER
        PARAMETER MIPC ACCESS TO SYSTEM C
        DDN  TESTDS2       TYPE GF
        PARAMETER 3
        DSMG0010I DISPLAY COMPLETE
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG DM–Dismount Data Set

Use this command to remove a data set from TPF system availability. You can dismount an entire data set or a specific volume of a data set.

Requirements and Restrictions

None.

Format

```
                                                   ┌─  ───────────────┐
                                                   │      (1)         │
►►──ZDSMG DM── ──ddname──┬────────────────────┬─ ──▼──┬─────────────────┬──┴──────────►◄
                         └─ ──DSName-dsname───┘       ├─Volume-number───┤
                                                      └─ALL─────────────┘
```

Notes:

1 If you specify both the ALL and VOLUME parameters, the VOLUME parameter must equal 1

ddname
: dismounts the specified data set, where *ddname* is the 1- to 16-character alphanumeric data definition name of the data set.

DSName-*dsname*
: dismounts the specified data set, where *dsname* is the 1- to 44-character alphanumeric data set name.

Volume-*number*
: dismounts a specific volume of the data set, where *number* is the volume sequence number from 1–99.

ALL
: dismounts the entire data set.

Additional Information

None.

Examples

The TESTDSN data set is dismounted in the following example.

```
User:   ZDSMG DM TESTDSN ALL

System: DSMG0004I 07.10.32 DDNAME TESTDSN         DISMOUNTED
```

The data definition name as well as the data set name is specified in the following example to dismount the TESTDSN data set.

```
User:   ZDSMG DM TESTDSN DSN-TPF.TEST.VOL01.MAY1889 V-2

System: DSMG0004I 07.10.32 DDNAME TESTDSN         DISMOUNTED
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG INIT–Initialize

Use this command to initialize general data set records during the next initial program load (IPL).

Requirements and Restrictions

None.

Format

```
►►──ZDSMG Init──────────────────────────────────────────────────────────►◄
```

Additional Information

The general data set records are initialized using the premount records.

Examples

In the following example, the general data set records are scheduled to be initialized during the next IPL.

```
User:   ZDSMG I

System: DSMG0036I 08.52.43 GDS INITIALIZATION SCHEDULED FOR NEXT IPL
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG MT–Mount Data Set

Use this command to make a data set available to the TPF system or to add a volume to a data set that is already mounted.

Requirements and Restrictions

None.

Format

```
►►──ZDSMG MT── ──address── ──ddname──┬──────────────────┬──────────────────────►
                                     └─ ──DSName-dsname─┘

   ┌─ ──Volume-1──────┐
►──┴─ ──Volume-number─┴───────────────────────────────────────────────────────►◄
```

address
: is a 3- to 4-digit hexadecimal device address.

ddname
: is a 1- to 16-character alphanumeric data definition name.

DSName-*dsname*
: is a 1- to 44-character alphanumeric data set name.

Volume-*number*
: mounts a specific volume of the data set, where *number* is the volume sequence number from 1–99.

Additional Information

- When you mount a data set, you must specify both the data definition name and the data set name unless they are the same. If the names are the same, specify only the data definition name.
- When you add a volume to a data set that is already mounted, specify only the data definition name. You can specify both the data definition name and the data set name, but they must agree with the values of the data set that is already mounted.

Examples

The TESTDSN data set is mounted in the following example. Notice that both the data definition name and data set name are specified.

```
User:   ZDSMG MT 03E6 TESTDSN DSN-TPF.SYSTEM.TEST.VOL01

System: DSMG0002I 07.10.32 DDNAME TESTDSN          VOLUME  1  03E6  MOUNTED
```

In the following example, a new volume is added the TESTDSN data set.

```
User:   ZDSMG MT 03E6 TESTDSN VOL-6

System: DSMG0002I 07.10.32 DDNAME TESTDSN          VOLUME  6  03E6  MOUNTED
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG RELEASE–Release Data Definition Name

Use this command to remove a data definition name from the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZDSMG RELease── ──ddname──────────────────────────────────────────►◄
```

ddname
is a 1- to 16-character alphanumeric data definition name.

Additional Information

If you remove a data definition name of a general data set, the entire general data set is dismounted as if you entered the ZDSMG DM command.

Examples

The TESTDSN data definition name is removed from the TPF system in the following example.

```
User:   ZDSMG REL TESTDSN

System: DSMG0104I 07.10.32 DDNAME TESTDSN          RELEASED
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSMG VTOC–VTOC Display

Use this command to display the volume table of contents (VTOC) for a data set.

Requirements and Restrictions

None.

Format

```
                              ┌─ ──ALL──────────┐
►►──ZDSMG Vtoc── ──address────┤                 ├──────────────────────►◄
                              └─ ──DSName-dsname┘
```

address
is a 3- to 4-digit hexadecimal device address.

ALL
displays the VTOC for all the data sets.

DSName-*dsname*
displays the VTOC for a specific data set, where *dsname* is the 1- to 44-character alphanumeric data set name.

Additional Information

If the device is not mounted when you enter this command, it is mounted to start the VTOC scan and dismounted when the scan is completed.

Examples

The VTOC for the TPF.TEST.G0002V00 data set is displayed in the following example.

```
User:   ZDSMG VTOC 03E1 DSN-TPF.TEST.G0002V00

System: DSMG0005I 07.10.32 TPF.TEST.G0002V00
        DSMG0006I 07.10.32 VOLUME NUMBER  1, NUMBER EXTENTS  7
```

The VTOC for all the data sets is displayed in the following example.

```
User:   ZDSMG VTOC 04E6

System: DSMG0005I 08.53.47 FMT381
        DSMG0005I 08.53.47 FMTFORK
        DSMG0005I 08.53.47 FMT1055
        DSMG0005I 08.53.47 MVS1055
        DSMG0005I 08.53.47 MVSFORK
        DSMG0005I 08.53.47 OSD381
        DSMG0005I 08.53.47 OSD1055
        DSMG0005I 08.53.47 OS3FORK
        DSMG0005I 08.53.47 OSDFORK
        DSMG0007I 08.53.48 VTOC SCAN COMPLETE
```

Related Information

See *TPF Database Reference* for more information about general data set support.

ZDSVC–Display SVC Code in Hexadecimal

Use this command to display the supervisor call (SVC) code for a specified macro and the index number of the entry in the secondary SVC table (if this is an indexed SVC). The information is displayed in hexadecimal format.

Requirements and Restrictions

None.

Format

```
►►──ZDSVC── ──macro──────────────────────────────────────────────►◄
```

macro
is a 5-character alphanumeric macro name or a 6-character alphanumeric mnemonic representing macros that generate different SVCs depending on the parameters supplied. Table 6 shows these mnemonics and the macros they represent.

Table 6. Macro Mnemonics for the ZDSVC Command

Mnemonic	Macro Name
CRETCM, CRETCS	CRETC
FILSCP, FILSCD	FILSC
FINSCP, FINSCD	FINSC
SENDCA, SENDCB, SENDCK, SENDCL, SENDCC, SENDCT	SENDC
KETSCL, KETSCS, KETSCD	KETSC
KETLCL, KETLCS, KETLCD	KETLC
PLONC1, PLONC2	PLONC

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZDSVC HELP
  ZDSVC ?
- If a macro name contains a dollar sign ($), you must specify 2 dollar signs ($$) instead of the single dollar sign. Single dollar signs entered as part of a macro name are interpreted by the message parser as a single quotation ('). For example, to display the hexadecimal SVC code for the $DCOLC macro, enter **ZDSVC $$DCOLC**.

Examples

The following example displays the hexadecimal SVC code for the FINWC macro.

```
User:   ZDSVC FINWC

System: DSVC0001I 09.59.56 FINWC  PRIMARY HEX 26
```

The following example displays the hexadecimal SVC code for the PROGC macro.

```
User:   ZDSVC PROGC

System: DSVC0004I 09.59.56 PROGC  PRIMARY HEX 3B  INDEX HEX 0000
```

Related Information

See *TPF Main Supervisor Reference* for more information about supervisor call (SVC) code.

ZDSYS–Display System Operating State

Use this command to display the current system operating state, which can be one of the following:
- 1052 state
- Utility (UTIL) state
- CRAS state
- Message switching state
- Normal (NORM) state.

Requirements and Restrictions

None.

Format

```
►►──ZDSYS──────────────────────────────────────────────────────────►◄
```

Additional Information

You can use the ZCYCL command to change the system operating state.

Examples

The following example displays the current system operating state.

```
User:   ZDSYS

System: DSYS0001I 14.36.25 THE SYSTEM IS IN NORM STATE
```

Related Information

See *TPF Main Supervisor Reference* for more information about system states.

ZDTAP–Display the Status of Tape Devices

Use this command to display the status of tape devices.

Requirements and Restrictions

None.

Format

```
>>──ZDTAP──┬──────────────────────────┬─┬──────────────────────┬──────────────><
           │    ┌─────────────────┐   │ ├──X───────────────────┤
           │    ▼                 │   │ ├──Y───────────────────┤
           └──────┬─address──────┬┴───┘ └─┬───────┬───┬───────┬┘
                  ├─/name────────┤        ├──G────┤   ├──I────┤
                  └─-group───────┘        └──S────┘   ├──O────┤
                                                      ├──R────┤
                                                      └──M────┘
```

address
: is the first 1–3 hexadecimal digits of a tape device address.

name
: is the first 1–3 alphanumeric characters of a tape name.

group
: is a 1- to 8-character alphanumeric tape group name.

X displays status information about all the tape devices that are currently configured. This parameter is provided primarily for diagnostic purposes.

Note: If no tapes are mounted on a tape device (that is, the tape device is available), the information displayed may not be valid. Tape name YYY is displayed for these available tape devices.

Y displays only the tape devices that do not have any tapes mounted.

M displays only the tape devices that have automatic tape mounting enabled and not suspended.

G displays only the tape devices where active tapes are mounted.

S displays only the tape devices where standby tapes are mounted.

I displays only the tape devices where input tapes are mounted.

O displays only the tape devices where output tapes are mounted.

R displays only the tape devices where real-time tapes are mounted.

Additional Information

- Enter **ZDTAP** (with no parameters) to display the status of all the defined tape devices.
- If `TAPE RESTART INCOMPLETE` is displayed when you enter this command, the TSTB verification is not completed.

Examples

The following information is displayed in the examples:

ADDRESS
is the logical address of the tape device.

NAME
is the name of the tape mounted on the tape device. If no tape is mounted, AVAIL is displayed or, if you specified the X parameter, YYY is displayed.

SSU
is the name of the subsystem user.

STATUS
is the status of the tape mounted on the tape device, where:

A Indicates an active tape.

S Indicates a standby tape.

I Indicates an input tape.

O Indicates an output tape.

* Indicates that the tape is being used or modified by the tape routines themselves. (This does not necessarily mean that an I/O operation is being performed.)

TPIND
is the contents of the primary, secondary, and tertiary status bytes in the tape status table (TSTB). When set, the status indicators have the following meaning.

Bit	Primary	Secondary	Tertiary
0	Inhibit I/O initiation	Reserved	Tape write immediate mode
1	Inhibit queue restart	Reserved	Tape is in compacted format
2	Tape is auxiliary queued	Standard user tape label on tape	Tape is non-spanned variable blocked format
3	Tape is a standby tape	Backward switch in progress	Tape device is enabled for automatic tape mounting
4	Tape is reserved	Reserved	Automatic tape mounting is suspended for the tape device
5	Reserved	Not first volume	Reserved
6	Tape is offline	Tape is an input tape	Reserved
7	Tape is closed	Tape is labeled	Reserved

VOLSER
is the volume serial number.

FORMAT
is the format of the tape.

#BLOCKS
is the number of blocks read from or written to the tape.

LDR
: indicates the presence of the automatic cartridge loader (ACL) feature. This field contains YES or NO.

The status of all the defined tape devices is displayed in the following example.

```
User:   ZDTAP

System: COTE0002I 08.27.43 DTAP      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         480      RTA    BSS      AO    00 01 30  A00049  38K           1 YES
         481     AVAIL

        END OF DISPLAY
```

All the available tape devices are displayed in the following example.

```
User:   ZDTAP Y

System: COTE0002I 10.39.40 DTAP      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         481     AVAIL

        END OF DISPLAY
```

In the following example, status information about the specified tape device is displayed whether that tape device is available or not.

```
User:   ZDTAP 481 X

System: COTE0002I 10.24.48 DTAP      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         481      YYY    BSS      AO    01 01 00  *NONE*  38K           5 YES

        END OF DISPLAY
```

All the tape devices where standby tapes are mounted are displayed in the following example.

```
User:   ZDTAP S

System: COTE0002I 08.31.44 DTAP      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         481      RTA    BSS      SO    11 01 30  A00050  38K           0 YES

        END OF DISPLAY
```

All the tape devices that were assigned to the specified tape group are displayed in the following example.

```
User:   ZDTAP -TEST1

System: COTE0002I 08.32.08 DTAP      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         480      RTA    BSS      AO    00 01 30  A00049  38K           1 YES

        END OF DISPLAY
```

Related Information

None.

ZDTCP–TCP/IP Connectivity Diagnosis Tools

Use this command to do the following:

- Define Domain Name System (DNS) server addresses
- Display DNS server addresses
- Display Transmission Control Protocol/Internet Protocol (TCP/IP) socket statistics
- Resolve TCP/IP host names and IP addresses
- Check the connectivity of a remote Internet Protocol (IP) host
- Display the route to an IP host.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
>>──ZDTCP── ──────────────────────────────────────────────────────────────────>

              ┌ ──────────────────┐
              ▼                   │
>──┬─DNS── ──┬─┬─PRImary-ipaddr─┬─┴─┬──────────────────────────┬────────────────><
   │         │ └─SECndry-ipaddr─┘   │                          │
   │         └─DISplay──────────────┘                          │
   ├─NETstat───────────────────────────────────────────────────┤
   ├─NSLookup── ──host─────────────────────────────────────────┤
   │                 ┌ ──TImeout-1──────┐                      │
   ├─PING── ──host───┼──────────────────┼──┬───────────────────┬┤
   │                 └ ──TImeout-second─┘  └ ──LOCal-iplocal───┘│
   └─TRACERte── ──host──┬───────────┬──┬───────────────────┬───┘
                        └ ──SMLCONS─┘  └ ──LOCal-iplocal───┘
```

DNS
: displays DNS server addresses.

PRImary
: defines the address of the primary DNS server to use for the `gethostbyname` and `gethostbyaddr` functions.

SECndry
: defines the address of the secondary DNS server to use for the `gethostbyname` and `gethostbyaddr` functions.

ipaddr
: is the numeric IP address of the DNS server.

DISplay
: displays the primary and secondary DNS server address. If an IP address is not defined for either the primary or secondary servers, NONE is displayed.

NETstat
: displays TCP/IP socket statistics.

NSLookup
displays the associated IP addresses for a specified host name or host name alias, or displays the host name for a specified IP address. If you specify a host name alias, the actual host name is displayed as well as the alias and associated IP addresses.

host
is the host name or IP address of a remote IP host. Specify the host name in alphanumeric characters. Each portion of the host name separated by dots must be no larger than 63 alphanumeric characters.

Note: Hyphens (-) are also allowed in the host name.

PING
checks the connectivity to a remote IP host. A response by the host indicates the host is available.

TImeout-*second*
waits a specified number of seconds for a pinged host to respond, where *second* is a number from 1–180.

LOCal-*iplocal*
specifies the local IP address, where *iplocal* is the numeric IP address. If you do not specify this parameter, the default local IP address is used. Do not enter the following IP addresses for the LOCAL parameter:

- 0.0.0.0
- 255.255.255.255
- 127.0.0.0

TRACERte
displays the route to an IP host.

In addition, network error indicators are displayed if they occur. These error indicators consist of the following:

***** Probe timed out.

! The time-to-live (ttl) is ≤ 1.

!H Host error.

!N Network error.

!P Protocol cannot be reached.

!S Source route failed.

!F Fragmentation needed.

SMLCONS
measures the output lines from the TRACERTE parameter and splits in the middle any line longer than 64 characters. The default is that the line is not split. Use the SMLCONS parameter when a console display does not permit lines more than 64 characters wide.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZDTCP HELP
ZDTCP ?

Examples

The following example defines a primary TPF DNS IP address.

```
User:   ZDTCP DNS PRIMARY-9.117.26.98

System: DTCP0005I 08.41.25 TPF DNS SERVER 9.117.26.98 ADDED TO DNS TABLE
```

The following example displays the TPF DNS IP addresses.

```
User:   ZDTCP DNS DISP

System: DTCP0006I 08.41.25 TPF DNS Server PRIMARY-9.117.26.98
                                          SECONDARY-9.117.25.19
```

The following example displays the TPF DNS IP address for the primary server, but the secondary server is not defined.

```
User:   ZDTCP DNS DISP

System: DTCP0006I 08.41.25 TPF DNS Server PRIMARY-9.117.26.98
                                          SECONDARY-NONE
```

The following example displays TCP/IP socket statistics. The first set of statistics shown is for TCP/IP offload support; the second set of statistics shown is for TCP/IP native stack support. The number of total sockets displayed at the bottom of the screen applies only to TCP/IP native stack support.

```
User:   ZDTCP NETSTAT

System: DTCP0003I 14.49.16 TPF TCP/IP NETSTAT
         SOCKET  TYPE  APPLICATION  LOCAL    COUNT  STATUS
                                     PORT
        -------  ----  -----------  -----  -------  ------
             33  UDP                   69  *******  SERVER
             34  TCP                 8000        2  LISTEN

        SOCKET   APP NAME   TYPE  LOCAL IP         LPORT  COUNT  STATUS
        -------- ---------- ----  ---------------  -----  -----  -----------
        00C000C9 RIP        UDP              ANY    520  *****  SERVER
        00C000CB TFTP       UDP              ANY     69  *****  SERVER
        00C000CA FTP        TCP              ANY     21      0  SERVER
        00C000CC MQ         TCP              ANY   1414      0  SERVER
        00C000CF FARES      TCP              ANY   5001      0  SERVER
                 1 RAW SOCKETS
                 2 UDP CLIENT SOCKETS
                 7 TCP CLIENT SOCKETS
                15 TOTAL SOCKETS
        END OF DISPLAY
```

The following example displays the host name for the specified IP address.

```
User:   ZDTCP NSL 129.42.19.99

System: DTCP0007I 09.24.08 NAME SERVER RESOLUTION DISPLAY

          NAME      - www.ibm.com
          ADDRESSES - 129.42.19.99

        END OF DISPLAY
```

The following example displays the host name and associated IP addresses for the specified host name alias.

```
User:   zdtcp nsl www.espn.go.com

System: DTCP0007I 09.24.08 NAME SERVER RESOLUTION DISPLAY

          NAME      - espn.go.com
          ADDRESSES - 204.202.131.230 204.202.129.230
          ALIASES   - www.espn.go.com

        END OF DISPLAY
```

The following example displays the ping of a TCP/IP station using a specific local IP address in numeric format.

```
User:   ZDTCP PING 9.117.33.9 LOCAL-9.117.31.5

System: DTCP0009I 11.56.54 PING STARTED
        DTCP0001I 11.56.54 PINGED HOST 9.117.33.9 (9.117.33.9)
                           RESPONSE TOOK 16 MSECS
```

The following example displays the ping of a TCP/IP host. In this example, the name server provides the host name (KGNVMC.POK.IBM.COM).

```
User:   ZDTCP PING KGNVMC.POK.IBM.COM

System: DTCP0009I 11.55.37 PING STARTED
        DTCP0001I 11.55.37 PINGED HOST 9.117.33.9 (KGNVMC.POK.IBM.COM)
                   RESPONSE TOOK 22 MSECS
```

The following example displays the route to an IP host using a specific local IP address in numeric format.

```
User:   ZDTCP TRACERTE 9.117.33.9 LOCAL-9.117.31.5

System: DTCP0004I 14.30.10 ROUTE TO 9.117.33.9 AS FOLLOWS
         1 9.117.198.1 (9.117.198.1)     10 ms   10 ms   9 ms
         2 9.117.197.251 (9.117.197.251)    *  7894 ms   9 ms   9 ms
         3 9.117.1.19 (9.117.1.19)     12 ms   10 ms   9 ms
         4 kgnvmc.pok.ibm.com (9.117.33.9)     14 ms   16 ms   14 ms
        END OF DISPLAY
```

The following example shows a host being unreachable because of networking problems (notice the network error indicators).

```
User:   ZDTCP TRACERTE 9.117.69.34

System: DTCP0004I 14.27.22 ROUTE TO 9.117.69.34 AS FOLLOWS
         1 9.117.198.1 (9.117.198.1)     8 ms   11 ms   11 ms
         2 9.117.197.252 (9.117.197.252)     21 ms   8 ms   11 ms
         3 9.117.1.1 (9.117.1.1)     9 ms   9 ms   9 ms
         4 9.117.1.1 (9.117.1.1)    *  7531 ms  *  8001 ms !N   5798 ms
         5 9.117.1.1 (9.117.1.1)    *  7637 ms  *  8000 ms  *  8000 ms
         6 9.117.1.1 (9.117.1.1)    *  7970 ms  *  8000 ms  *  8000 ms
         7 9.117.1.1 (9.117.1.1)    *  7968 ms  *  7999 ms  *  8003 ms
         8 9.117.1.1 (9.117.1.1)   !N   3196 ms  *  7763 ms !N   5265 ms
        UNREACHABLE
        END OF DISPLAY
```

The following example displays how the use of the SMLCONS parameter splits lines longer than 64 characters.

```
User:   ZDTCP TRACERTE 9.117.69.34 SMLCONS

System: DTCP0004I 18.38.09 ROUTE TO 9.117.69.34 AS FOLLOWS
         1 9.117.197.251 (9.117.197.251)
         * 5017 ms  * 5004 ms  * 5060 ms
         2 9.117.1.1 (9.117.1.1)    58 ms  50 ms  49 ms
         3 pkmfgvm4.pok.ibm.com (9.117.32.15)
         * 5013 ms  * 5006 ms  * 4995 ms
         4 9.117.69.34 (9.117.69.34)   * 5008 ms   25 ms   22 ms
        END OF DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP support.

ZDTIM–Display Time

Use this command to display the local standard time (LST) for the TPF system or subsystems, or the current time and characteristics of the Sysplex Timer.

Requirements and Restrictions

None.

Format

```
►►──ZDTIM──┬───────────────────┬──────────────────────────────────────►◄
           ├──SYS──────────────┤
           ├──ALL──────────────┤
           ├──TOD──────────────┤
           └──STR──┬─────────┬─┘
                   └─ALTPORT─┘
```

SYS
displays only the system clock.

ALL
displays the system clock and all subsystem clocks that are above 1052 state.

TOD
displays the current value of the TOD clock as well as clock synchronization information.

STR
displays the current time and characteristics of the Sysplex Timer. This parameter is valid only for a central processing complex (CPC) that is directly connected to a Sysplex Timer.

If the CPC is in Sysplex Timer stepping mode, the active stepping port is used. If the CPC is not in Sysplex Timer stepping mode, the port that would become the active stepping port is used.

Note: Retrieving data from the Sysplex Timer may take several seconds. If the TOD clock was set to use the Sysplex Timer time, enter **ZDTIM TOD** to display the time.

ALTPORT
displays the current time and characteristics of the Sysplex Timer on the port that is not the active stepping port. This parameter is valid only with the Sysplex Timer and is not allowed when running under Processor Resource/Systems Manager (shared PR/SM).

Additional Information

- Enter **ZDTIM** (with no parameters) to display the subsystem local standard time. Use this time as the base time for changing the system time-of-day (TOD) clock.
- The system clock is not running when the basic subsystem (BSS) is in 1052 state.

- When the TPF system is in 1052 state and a time-of-day (TOD) message is generated, the hours and minutes portion of the TOD clock message is stored instead of the LST message.
- When you display the subsystem LST clock and the subsystem is in 1052 state, the TPF system calculates the subsystem LST based on the system clock and the time difference between the system clock and the subsystem clocks.
- Use the system local standard time that is displayed when you specify the TOD or STR parameters as the base for changing the system time-of-day (TOD) clock.
- The Sysplex Timer applies the same Greenwich mean time (GMT) offset as the TOD clock.
- Because the ZDTIM STR command displays the Sysplex Timer time value and the TOD value, use the TOD clock value as the verification time based on the ZATIM command.
- The TPF system uses the hours and minutes (*hh.mm.*) portion of the LST message for verification if you enter a subsequent request to change the system or subsystem time.
- If there is no master processor in the complex, the master processor ID and serial number are not displayed.

Examples

The following example displays the local standard time for the BSS.

```
User:   ZDTIM

System: DTIM0001I 14.55.15 SUBSYSTEM BSS  LOCAL STANDARD TIME
```

The following example displays the current value of the TOD clock.

```
User:   ZDTIM TOD

System: CLKS0010I 13.34.05 TIME OF DAY CLOCK LOCAL STANDARD TIME
                           TIME: 13.34.05  DATE: 05/23/94
        CLKS0081I 13.34.05 SOURCE: MASTER  STATUS: CONFIRMED
                           MASTER- CPUID: B  SERIAL: 020410  MODEL: 3090
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Generation* for more information about the system clocks.

ZDTLB–Display Tape Label

Use this command to display a tape label.

Requirements and Restrictions

None.

Format

```
►►──ZDTLB── ──tape──────────────────────────────────────────────►◄
```

tape
is a 3-character tape name. The first 2 characters are alphabetic and the 3rd character is alphanumeric.

Additional Information

None.

Examples

The tape label for the specified tape is displayed in the following example, where:

USABLE
indicates the tape status. If this field displays NOT USABLE, the label information is not valid until you make it usable by entering the ZTLBL command with the USABLE parameter.

I/O
indicates whether the tape is an input tape, an output tape, or both.

L
is the label type, where:

SL
Tape has standard header labels.

NL
Tape does not have standard header labels when used for input.

SU
Tape has standard header labels followed by user labels when used for output.

FMT
indicates the format of the tape.

COMP
indicates the improved data recording capability (IDRC) mode of the tape, which can be COMP or NOCOMP.

BLK
indicates the blocking mode of the tape, which can be BLK or NOBLK.

T
is the buffer time delay, which indicates the number of seconds the data can reside in the blocking buffer before a synchronize operation is scheduled.

Note: If 000 is displayed, the tape is not subject to time-initiated buffer synchronization.

F
: is the file identifier.

G
: is the current generation number.

S
: is the current volume sequence number.

RETENTION PERIOD
: is the retention period.

LAST MOUNTED
: is the date that the tape was last mounted.

LAST FILE SERIAL NUMBER
: is the volume serial number of the first volume of the last file that was mounted.

GROUP
: is the tape group name that is associated with the label.

Note: If a tape has an NL tape label, only the first 2 lines of output are displayed.

```
User:   ZDTLB RTA

System: COTL0002I 08.32.08 DTAP      - TAPE STATUS
           RTA - USABLE        I/O-OUTPUT  L-SL   FMT-ALL  NOCOMP
           BLK                 T  - 000
           F-RTA.TAPE          G  -0027    S-0001
           RETENTION PERIOD -  00007
           LAST MOUNTED     -  94194
           LAST FILE SERIAL -  A00049
           GROUP            -  NONE
```

Related Information

None.

ZDTOD–Display Date, Time, and Time-Of-Day (TOD) Clock

Use this command to display:

- The current value of the time-of-day (TOD) clock, including a translation to a date and time. The TOD clock is set to Greenwich Mean Time (GMT).
- The corresponding date and time for a specific TOD clock value.
- The corresponding TOD clock value for a specific date and time.

Requirements and Restrictions

- You can display the date only up to the year 2041 with this command. That is, use of this command is valid only between January 1, 1900 00:00:00 (midnight) and December 31, 2040 23:59:59.
- The date format you can specify with this command is determined by the format coded on the DATEFMT parameter of the CONFIG macro. For example, if DATEFMT=MDY is coded on the CONFIG macro, you can only specify a date using the *mm/dd/yy* and *mm/dd/yyyy* date formats. See *TPF System Generation* for more information about the date formats and the CONFIG macro.

Format

```
►►──ZDTOD──┬─────────────┬──────────────────────────────────────────►◄
           ├─ ──todclock─┤
           └─ ──datetime─┘
```

todclock
is a TOD clock value from 1–16 hexadecimal digits.

datetime
is a date and time in one of these formats:
- *mm/dd/yy*
- *mm/dd/yyyy*
- *dd/mm/yy*
- *dd/mm/yyyy*
- *yy/mm/dd*
- *yyyy/mm/dd*
- *hh.mm.ss*

where:

mm
is the month.

yy is the year when it is less than the year 2000.

yyyy
is the year when it is less than the year 2041 and greater than 1999.

dd is the day.

hh is the hour.

mm
is the minute.

ss is the second.

Additional Information

- Enter **ZDTOD** (with no parameters) to display the current date and time, as well as the current value of the TOD clock.
- The default display of the current system date and time is *mm*/*dd*/*yy* when the year specified is less than 2000 and *mm*/*dd*/*yyyy* when the year specified is less than 2041 and greater than 1999.

  To change the default to another date format, you must modify the DATEFMT parameter on the CONFIG macro. See *TPF System Generation* for more information about the CONFIG macro.

Examples

The following example displays the current date, time, and TOD clock value when you enter ZDTOD without any parameters.

```
User:   ZDTOD

System: CSMP0097I 21.46.06 CPU-B SS-BSS   SSU-HPN   IS-01
        DTOD0001I 21.46.06 DATE          TIME          TODCLOCK
                CURRENT  1996/10/02    18.25.19    AD8EEB52 0810F241
```

The following example displays the date, time, and TOD clock value when you specify a specific date.

```
User:   ZDTOD 1996/10/02 18.25.19

System: CSMP0097I 21.46.06 CPU-B SS-BSS   SSU-HPN   IS-01
        DTOD0002I 21.46.06 DATE          TIME          TODCLOCK
                INPUT    1996/10/02    18.25.19    AD8EEB51 561C0000
```

The following example displays the date, time, and TOD clock value when you specify a specific TOD clock value.

```
User:   ZDTOD AD8EEB51561C0000

System: CSMP0097I 21.46.06 CPU-B SS-BSS   SSU-HPN   IS-01
        DTOD0002I 21.46.06 DATE          TIME          TODCLOCK
                INPUT    1996/10/02    18.25.19    AD8EEB51 561C0000
```

Related Information

See *TPF Main Supervisor Reference* and *TPF System Generation* for more information about the system clocks.

ZDUMP–Real-Time Manual Dump

Use this command to start a control program (CP) manual system error storage dump from a valid CRAS. The CP dump can be a selective dump, which includes only specific storage areas that are identified using the ZIDOT command, or a complete storage memory dump.

Requirements and Restrictions

None.

Format

```
           ┌ ──SEL─┐
►►──ZDUMP──┤       ├──────────────────────────────────────────────────────►◄
           └ ──ALL─┘
```

SEL
: starts a CP dump of a selected portion of system storage. The areas of storage are defined by the IDOTB macro and can be overridden using the ZIDOT command.

ALL
: activates a CP dump of all permanent storage and all main storage blocks.

  Attention: Use this options with extreme care. A full storage dump is time consuming and causes a large spike in system activity and resource usage when the dump ends.

Additional Information

- This command does not cause the TPF system to stop.
- The following system error options are overridden for the duration of the manual dump:
  - DUMPOFF option is forced to DUMPON.
  - SHORT option is forced to LONG.
  - System error message routing is forced to the prime CRAS.

Examples

A dump of selected areas of main storage is requested in the following example.

```
User:   ZDUMP

System: CPSE0011I 14.26.27 MANUAL DUMP IN PROGRESS, LOCATION 0015D5A0
        DUMP0000I 14.26.27 ZDUMP-OK
        CPSE0015I 14.26.27 IS-0001 SS-BSS  SSU-HPN  SE-000658 MANUAL DUMP
        010000B CVFM40  00D6
```

Related Information

See *TPF Main Supervisor Reference* for more information about system error dumps.

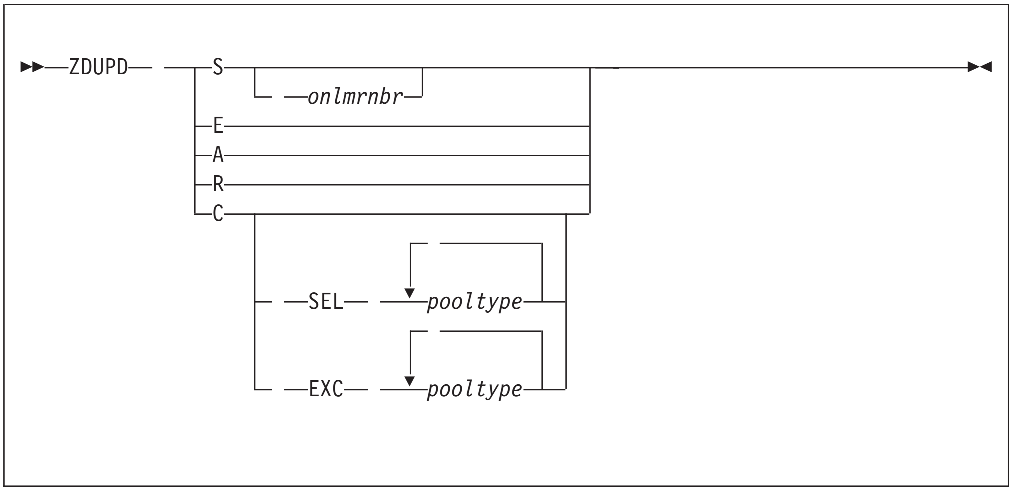
ZDUPD–Verify and Continue PDU Rollin

Use this command to verify, restart, roll in, or end pool directory update (PDU) processing.

Requirements and Restrictions

You can enter this command only after ZRPDU CREATE command processing has completed successfully.

Format



S verifies pseudo directories against the pool rollin directory (#SONRI) from multiple releases.

Note: If you run verify processing in 1052 state, the number of online multiple releases are not reported correctly.

onlmrnbr
is the number of online multiple releases per pool directory that are logged to the online multiple release database. Specify a 2-digit number from 01 to 99.

E displays all online multiple releases that were found during the verify step (ZDUPD S). A maximum of 1000 multiple releases can be displayed.

Notes:
1. You can enter this parameter only if the TPF Database Facility (TPFDF) product is installed.
2. You can enter this parameter only if the TPF system is in NORM state.

A forces ZDUPD command processing to end.

Notes:
1. If you force verify processing (ZDUPD S) to end, you can run it again by entering the ZDUPD command again with the S parameter specified.
2. If you force rollin processing (ZDUPD C) to end, you cannot restart it. Enter **ZRPDU ABORT BP** to force PDU processing to end without completing, and start online PDU processing again by entering the ZRPDU CREATE command.

R restarts rollin processing if the system has an unscheduled outage.

C continues with the rollin process after verify processing has been completed.

SEL
selects pool types to include in the rollin process.

pooltype
is a valid type of pool file. You can specify as many as five pool types.

EXC
selects pool types to exclude from the rollin process.

Additional Information

None.

Examples

The following example verifies pseudo directories against the pool rollin directory (#SONRI) and continues the rollin process after verify processing has been completed.

```
User:   ZDUPD S

System: DYDU0006I 09.38.19 -START OF DIRECTORY UPDATE VERIFY+
        CSMP0097I 09.38.19 CPU-B SS-BSS  SSU-HPN  IS-01
        DYDE0000I 09.38.19 DYDU - END OF DUPL CHECK PASS

        MULTIPLE RELEASES

        POOL  DEV
        TYPE  TYPE                      COUNTS
        ----  ----  --------------------------
        SDP   DEVA                           2
        LDP   DEVA                           2
        4DP   DEVA                           0
              DEVB                           0
        4D6   DEVA                           0
              DEVB                           0

        ENTER:       ZDUPD C - TO CONTINUE

         OR          ZDUPD A - TO ABORT+

User:   ZDUPD C

System: BCPY0001I 09.39.30  - KEYPOINT 9 AND SONRI SAVE STARTED
        CSMP0097I 09.39.30 CPU-B SS-BSS  SSU-HPN  IS-01
        BCPY0002I 09.39.30  - KEYPOINT 9 AND SONRI SAVE COMPLETE
        CSMP0097I 09.39.30 CPU-B SS-BSS  SSU-HPN  IS-01
        DYDU0098I 09.39.30 - START OF DIRECTORY UPDATE ROLLIN
        CSMP0097I 09.39.31 CPU-B SS-BSS  SSU-HPN  IS-01
        BCP20008I 09.39.31 DIRECTORY UPDATE/KEYPOINT 9 UPDATE COMPLETE
        CSMP0097I 09.39.31 CPU-B SS-BSS  SSU-HPN  IS-01
        BCPE0010I 09.39.31 BCPU - END OF DIRECTORY UPDATE

        ADDRESSES RETURNED

        POOL  DEV
        TYPE  TYPE                      COUNTS
        ----  ----  --------------------------
        SDP   DEVA                       9 264
        LDP   DEVA                      13 139
        4DP   DEVA                       8 324
              DEVB                       4 693
        4D6   DEVA                           0
              DEVB                       7 956
        BCPE0011I 09.39.31 BCPE - END OF DIRECTORY UPDATE
        CSMP0097I 09.39.31 CPU-B SS-BSS  SSU-HPN  IS-01
        DFPC0011I 09.39.31 21MAR CYD0  AVAILABLE FILE POOL COUNTS
                                        FILE      CORE      ORD
        SST DEVA                     190 630       763 00000000
        SDP DEVA                     133 046      1899 00000029
        LST DEVA                      68 838       462 0000003C
        LDP DEVA                     110 725     11251 0000005E
        4ST DEVA                      25 608       240 0000006E
            DEVB                       8 568        24 00000098
            TOT                       34 176       264
        4DP DEVA                      91 623     10124 0000008B
            DEVB                       4 693         0 000000B0
            TOT                       96 316     10124
        4D6 DEVA                         276        48 000000B4
            DEVB                      13 738     15955 000000C2
            TOT                       14 014     16003
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZDVSN–Display Volume Serial Number

Use this command to display the volume serial number of online or offline DASD.

Offline DASD is a DASD that is physically mounted and in ready state, but is not already mounted for use by the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZDVSN── ──ccud──────────────────────────────────────────────►◄
```

ccud
is the hardware address of the DASD.

Additional Information

None.

Examples

The volume serial number for the specified DASD is displayed in the following example.

```
User:   ZDVSN 03E0

System: DVSN0000I 09.00.24 VSN ON DISK 0EE5 IS BP0001
```

Related Information

See *TPF Database Reference* for more information about DASD support.

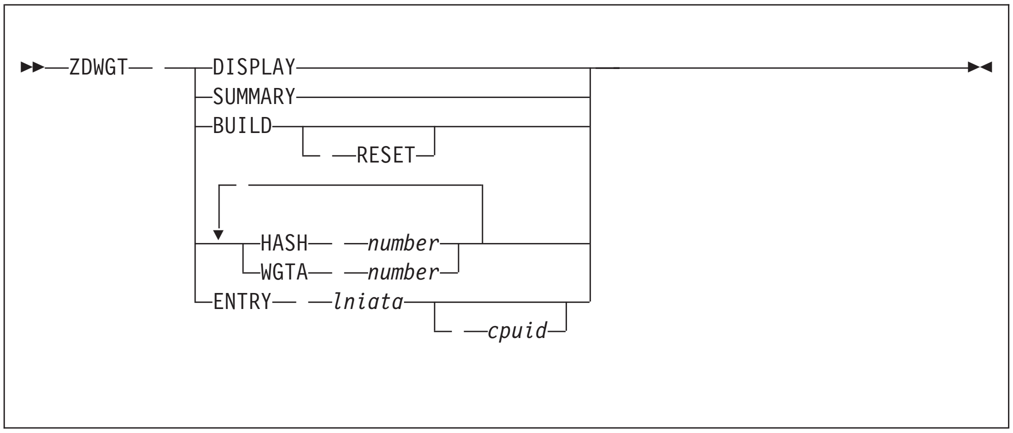
ZDWGT–Display or Modify the WGTA

Use this command to display or modify the information in the terminal control table (WGTA), which is located in keypoint C.

Requirements and Restrictions

None.

Format



DISPLAY
displays the number of hash table entries and WGTA entries in keypoint C.

SUMMARY
prints the hash table distribution for the WGTA.

BUILD
sets the WGTA BUILD switch in keypoint C. The BUILD switch indicates that the WGTAs are reinitialized during the next system restart. The previous WGTA is examined for entries that contain a nonzero RID value. Data (RID, application name table index, terminal type, terminal status, and 2 indicator bytes) from these entries is moved from the old table to the new one, saving it across the build.

RESET
sets the RESET switch in keypoint C to prevent previous data from being used. The WGTA entries are reset to their original values based on the user accepted test (UAT).

HASH
changes the number of hash table entries in keypoint C and sets the BUILD switch. The RESET switch is not set.

WGTA
changes the number of WGTA entries in keypoint C and sets the BUILD switch. The RESET switch is not set.

number
is a 1- to 6-digit decimal number of hash table or WGTA entries. For hash table entries, the number must be an odd number. For WGTA entries, the number must be greater than 1.

ENTRY
displays the WGTA entry for the line number, interchange address, and terminal address (LNIATA) and CPU ID specified.

lniata
is a 6-digit hexadecimal number for the line number, interchange address, and terminal address (LNIATA).

cpuid
is the 1-character alphanumeric CPU ID of the processor.

Additional Information

None.

Examples

The number of hash table entries and WGTA entries is displayed in the following example.

```
User:   ZDWGT DISPLAY

System: CGT40099I NUMBER OF HASH TABLE ENTRIES 001301
                  NUMBER OF WGTA TABLE ENTRIES 001300
```

The hash table distribution for the WGTA is displayed in the following example.

```
User:   ZDWGT SUMMARY

System: CGT30031I HASH TABLE DISTRIBUTION
        SLOTS WITH 0 ENTRIES 000642
        SLOTS WITH 1 ENTRY   000517
        SLOTS WITH 2 ENTRIES 000127
        SLOTS WITH 3 ENTRIES 000014
        SLOTS WITH 4 ENTRIES 000001
        NO SLOTS HAVE MORE THAN 4 ENTRIES.
        CGT40007I REQUEST COMPLETE
```

In the following example, the BUILD switch for the WGTA is set in keypoint C.

```
User:   ZDWGT BUILD

System: CGT40007I REQUEST COMPLETE
```

The number of hash table entries in keypoint C is changed in the following example.

```
User:   ZDWGT HASH 643

System: CGT40007I REQUEST COMPLETE
```

In the following example, information in the WGTA about LNIATA 000000 attached to CPU B is displayed.

```
User:   ZDWGT ENTRY 000000 B

System: CSMP0097I 13.26.55 CPU-B SS-BSS  SSU-HPN  IS-01
        CGTB0002I 13.26.55
        CPU ID           -  B
        FILE ADDRESS     -  FC02C005
        ORDINAL NUMBER   -      1
        LU IDENTIFIER    -  000000
        TERMINAL TYPE    -  A5
        ANT INDEX        -  95
        TA STATUS        -  60
        INDICATOR BYTES  -  02 C0 18
        USER FIELD       -  00 00 00 00
        END OF DISPLAY
```

Related Information

See *TPF Data Communications Services Reference* for more information about the WGTA.

ZECBL–Display, Remove, and Unsuspend ECBs

Use this command to do the following:

- Display information about:
  - Looping entry control blocks (ECBs)
  - Hung ECBs
  - Suspended ECBs
- Remove ECBs from storage
- Unsuspend suspended ECBs.

Requirements and Restrictions

None.

Format

```
►►──ZECBL ───┬─D── ───┬─LOOPing───┬──┬──────────────────────────────────────────►◄
             │        ├─HUNG──────┤  │
             │        ├─ALL───────┤  │
             │        ├─SUSPended─┤  │
             │        └─address───┘  │
             ├─E── ──address─────────┤
             └─U── ──address─────────┘
```

D displays information about the looping and hung ECBs.

LOOPing
displays information about the looping ECBs in the TPF system.

HUNG
displays information about the hung ECBs in the TPF system.

ALL
displays information about all the looping and hung ECBs in the TPF system.

SUSPended
displays information about suspended ECBs in the TPF system. These ECBs have been suspended because they ran a LODIC or TMSLC macro call.

address
is the system virtual address (SVA) of an ECB.

E removes an ECB from the TPF system.

U changes an ECB that was previously labeled as capable of being suspended to run without being suspended.

Note: This parameter is valid only for ECBs that issued the LODIC or TMSLC macros. It is equivalent to running a LODIC macro with the UNMARK parameter, or a TMSLC macro with the DISABLE parameter.

Additional Information

None.

Examples

The following example displays information about all looping ECBs.

```
User:   ZECBL D LOOPING

System: ECBL0021I 14.26.27   0021 ECB'S IN USE , 0000 LOOPING, 0000 HUNG
```

The following example displays information about a suspended ECB at a specified address.

```
User:   ZECBL D 00458000

System: ECBL0024I 10.46.00 ECB DATA DISPLAY

          SUSPENDED ECB AT 00458000, SS - BSS, SSU - HPN
            PROG COSA, AT 010ECB50, PSW 8103CE36,
            I/O COUNT   1, HOLD COUNT   1, ECB LIFE  0, MAX    0 MINUTES

        END OF ZECBL DISPLAY
```

The following example unsuspends a suspended ECB.

```
User:   ZECBL U 00458000

System: ECBL0014I 10.46.00 SUSPENDED ECB AT 00458000 CAN NOW BE DISPATCHED
```

Related Information

- See *TPF Concepts and Structures* for more information about ECBs.
- See *TPF General Macros* for more information about the LODIC and TMSLC macros.

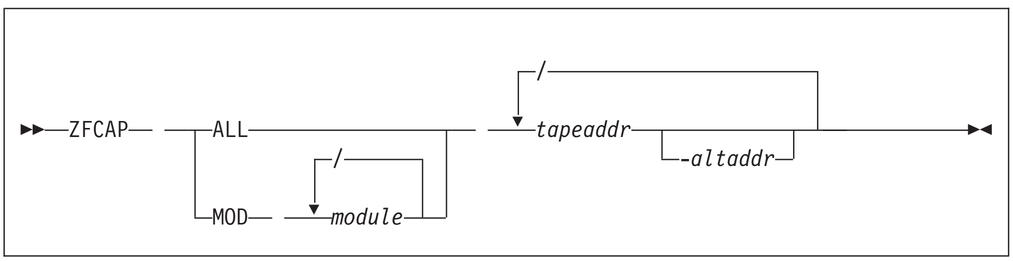
ZFCAP–Start the Capture Function

Use this command to start the capture function. You can capture all of the online modules (physical DASD units), or only specific online modules.

Requirements and Restrictions

- Specify at least one tape device for each participating processor.
- Load the capture tapes (BXA–BXZ, BX0–BX5), initialize them, and ensure that they are ready to use.
- Use the ZTMNT command to mount an exception tape (RTT–RTY) for each active processor, whether the capture function affects the processor or not.
- Each participating processor must be in UTIL state or higher.
- An entry for the capture function (CAP*x*, where *x* is an identifier that is unique for each processor) must exist and be active in the processor resource table (PROT) for each participating processor. Use the ZPROT command to define the PROT entry.
- The label directory definitions (BXA–BXZ, BX0–BX5) must define the capture tapes as both input and output tapes when the capture function is started. Otherwise, the restore function cannot mount the capture tapes.
- Do not use the ALL prefix with this command.

Format



ALL
captures all of the online modules.

MOD
captures one or more specific online modules.

module
is a 3-digit hexadecimal symbolic module number.

tapeaddr
is the 3-digit hexadecimal address of a primary tape device.

altaddr
is the 3-digit hexadecimal address of an alternate tape device. When specified, the capture function can alternate between the primary and alternate tape device.

Note: The addresses of the primary and alternate tape devices must be on the same device type.

Additional Information

- The capture function also starts exception recording on all active processors in the complex.

- The following are ***not*** captured when you capture all of the online modules:
  – Program area
  – Duplicate copies of records.
- Primary and alternate tape devices can be connected to different control units; however, this can decrease performance during the capture function.
- Use the ZFRST CAP command to restore the online modules.
- Use the ZFRST XCP command to restore the exception records.

Examples

The capture function is started for all of the online modules in the following example. Two primary tape devices are specified, and one of the primary tape devices has an alternate tape device.

```
User:   ZFCAP ALL 423/424-425

System: FCAP0021I 14.24.55 TAPE 423 ADDED
        FCAP0119I 14.24.55 DUAL TAPE 424-425 ADDED
        FCAP0010I 14.24.55 STARTED
        FCAP0008I 14.24.56 MOD 047 ON 0EE5 TO TAPE ON 423 - STARTED
        COTM0046I 14.24.56 TMNT HPN    TAPE BXA MOUNTED ON DEVICE 423
                           VSN A00064 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0112I 14.26.11 TAPE BXA REPLACED BY TAPE 424
        COTM0046I 14.26.11 TMNT HPN    TAPE BXA MOUNTED ON DEVICE 424
                           VSN A00065 G     S     D38K   SL  NOBLK  NOCOMP
        COTS0382I 14.26.11 TPSW HPN    TAPE BXA SWITCHED FROM 423 TO 424
                           VSN IS NOW A00065
        COTS0080A 14.26.11 TWEV HPN    REMOVE BXA FROM DEVICE 423
                           VSN A00064 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0009I 14.27.10 MOD 047 ON 0EE5 TO TAPE ON 424 - COMPLETED
        COTC0080A 14.27.10 TCLS HPN    REMOVE BXA FROM DEVICE 424
                           VSN A00065 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0134I 14.27.11 PROCESSOR B COMPLETED
        FCAP0011I 14.27.11 COMPLETED
        FCAP0044I 14.27.11 EXCEPTION RECORDING CONTINUING
        FCAP0001I 14.27.10
        CAPTURE STARTED 23MAY 1424
        MODULES TO DO        1
        MODULES COMPLETED    1
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE   Y
        IOTIME                004
        DISK ERRORS
        TRKS 000
        RCDS 000

            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   424   425   C0              B
          BXB   423         40              B
        END OF STATUS DISPLAY
```

The capture function is started for module 047 in the following example. A primary and alternate tape device pair is used.

```
User:   ZFCAP MOD 047 423-424

System: FCAP0057I 10.26.35 MOD 047 ADDED
        FCAP0119I 10.26.35 DUAL TAPE 423-424 ADDED
        FCAP0010I 10.26.35 STARTED
        FCAP0008I 10.26.35 MOD 047 ON 0EE5 TO TAPE ON 423 - STARTED
        COTM0046I 10.26.37 TMNT HPN    TAPE BXA MOUNTED ON DEVICE 423
                               VSN A00066 G     S     D38K   SL  NOBLK  NOCOMP
        COTM0046I 10.29.10 TMNT HPN    TAPE BXA MOUNTED ON DEVICE 424
                               VSN A00067 G     S     D38K   SL  NOBLK  NOCOMP
        COTS0382I 10.29.10 TPSW HPN    TAPE BXA SWITCHED FROM 423 TO 424
                           VSN IS NOW A00067
        COTS0080A 10.29.11 TWEV HPN    REMOVE BXA FROM DEVICE 423
                               VSN A00066 G     S     D38K   SL  NOBLK  NOCOMP
        COTC0080A 10.30.26 TCLS HPN    REMOVE BXA FROM DEVICE 424
                               VSN A00067 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0009I 10.30.26 MOD 047 ON 0EE5 TO TAPE ON 424 - COMPLETED
        FCAP0134I 10.30.26 PROCESSOR B COMPLETED
        FCAP0011I 10.30.26 COMPLETED
        FCAP0044I 10.30.26 EXCEPTION RECORDING CONTINUING
        FCAP0001I 10.30.26
        CAPTURE STARTED 14FEB 1026
        MODULES TO DO        1
        MODULES COMPLETED    1
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE   Y
        IOTIME                004
        DISK ERRORS
        TRKS 000
        RCDS 000

            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   424   423   C0              B
        END OF STATUS DISPLAY
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP–Start or Stop Exception Recording

Use this command to:

- Start exception recording for a processor that was not active when the capture function was started
- End exception recording for all active processors after the capture function is completed.

Requirements and Restrictions

- You can enter the ZFCAP EXCEP command only when you are prompted by the TPF system while you are cycling above 1052 state.
- Use the ZTMNT command to mount an exception tape (RTT–RTY) if you are starting exception recording for a processor.
- Use the ZTMNT command to mount a keypoint capture (KPC) tape if you are ending exception recording after the capture function is completed for all the online modules (ZFCAP ALL). In this case, the keypoint capture function starts automatically when you end exception recording. (It is not started automatically if only specific modules were captured.)
- Do not use the ALL prefix with this command.

Format

```
►►──ZFCAP──  ──┬─EXCEP─┬──────────────────────────────────────────────────►◄
               └─SEXPT─┘
```

EXCEP
starts exception recording for the current processor.

SEXPT
ends exception recording for all active processors.

Additional Information

- When you end exception recording, information about the final status of the capture function is displayed.
- The ZFCAP SEXPT command also starts an RTA tape switch.
- Use the ZFRST XCP command to restore the exception records.

Examples

Exception recording is started for the current processor in the following example.

```
User:   ZCYCL UTIL

System: BXAR0007I 15.23.15 START XCP RECORDING

User:   ZFCAP EXCEP

System: FCAP0136I 15.23.54 EXCEPTION RECORDING STARTED
        CYCL0001I 15.23.54 CYCL TO UTIL - STARTED
        CVCX0001I 12.39.55 SS BSS  NOW IN UTIL STATE
```

Exception recording is ended for all active processors in the following example. Information about the final status of the capture function is also displayed and the keypoint capture function is started.

```
User:   ZFCAP SEXPT

System: FCAP0038I 10.15.34 XCP STOPPED - RTA SWITCH INITIATED
        COTS0382I 10.15.34 TPSW BSS    TAPE RTA SWITCHED FROM 420 TO 421
                           VSN IS NOW A00070
        FCAP0001I 10.15.34
        CAPTURE STARTED 14FEB 1006
        MODULES TO DO        1
        MODULES COMPLETED    1
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED
        CURRENT MAXIMUM ECB
        CURRENT ACTIVE ECB
        XCP RECORDING ACTIVE
        DISK ERRORS
        TRKS 000
        RCDS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
         NO TAPES AVAILABLE
        END OF STATUS DISPLAY
        COTS0080A 10.15.34 TWEV BSS    REMOVE RTA FROM DEVICE 420
                               VSN A00274 G0012 S0001 D38K   SL  BLK    NOCOMP
        FCAP0025I 10.15.34 KPT STARTED
        COTG0080A 10.15.34 TOFF BSS    REMOVE RTU FROM DEVICE 422
                               VSN A00065 G0001 S0001 D38K   SL  NOBLK  NOCOMP
        FCAP0046I 10.15.38 KEYPOINT 1 NOT CAPTURED
        FCAP0046I 10.15.38 KEYPOINT 6 NOT CAPTURED
        FCAP0046I 10.15.38 KEYPOINT I NOT CAPTURED
        FCAP0027I 10.15.40 KPT COMPLETE
        COTC0080A 10.15.40 TCLS HPN    REMOVE KPC FROM DEVICE 425
                               VSN A00068 G0001 S0001 D38K   SL  NOBLK  NOCOMP
```

Related Information

See *TPF Database Reference* for more information about exception recording.

ZFCAP ABORT–Abort the Capture Function

Use this command to immediately end the capture function or to temporarily stop capturing one or more specific online modules (physical DASD units).

Requirements and Restrictions

- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format

```
►►──ZFCAP ABORT── ──┬─ALL──────────────────────┬─────────────────────────────►◄
                    │            ┌─/──────┐    │
                    │            ▼        │    │
                    └─MOD── ─────module───┴────┘
```

ALL
ends the capture function for all of the online modules that are being captured.

MOD
temporarily stops capturing one or more specific online modules.

module
is a 3-digit hexadecimal symbolic module number of a DASD module that is being captured.

Additional Information

- If you temporarily stop capturing one or more specific online modules, the modules are automatically captured again at a later time.
- Exception recording and record logging continue if they were running.

Examples

The capture function is ended for all the online modules in the following example.

```
User:   ZFCAP ABORT ALL

System: FCAP0042I 10.57.08 ABORT IN PROGRESS
        FCAP0016I 10.57.08 MOD 047 ON 0EE5 TO TAPE ON 423 - ABORTED
        FCAP0015I 10.57.08 ABORTED
        FCAP0044I 10.57.08 EXCEPTION RECORDING CONTINUING
        COTC0080A 10.57.11 TCLS HPN    REMOVE BXA FROM DEVICE 423
                               VSN A00066 G     S     D38K   SL  NOBLK  NOCOMP
```

The capture function temporarily stops capturing module 047 in the following example.

```
User:   ZFCAP ABORT MOD 047

System: FCAP0016I 11.08.26 MOD 047 ON 0EE5 TO TAPE ON 423 - ABORTED
        COTC0080A 11.08.26 TCLS HPN    REMOVE BXA FROM DEVICE 423
                               VSN A00066 G     S     D38K   SL  NOBLK  NOCOMP
```

The TPF system temporarily stops capturing modules 01E and 01F in the following example.

```
User:   ZFCAP ABORT MOD 01E/01F

System: COTC0080A 10.16.13 TCLS WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0016I 10.16.13 MOD 01E ON 03E7 TO TAPE ON 423 - ABORTED
        FCAP0016I 10.16.13 MOD 01F ON 0EEB TO TAPE ON 480 - ABORTED
        COTC0080A 10.16.13 TCLS WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248 G     S     D38K   SL  NOBLK  NOCOMP
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP ALTER–Change the Number of Simultaneous Captures Allowed

Use this command to change the number of captures allowed at one time onto current processor.

Requirements and Restrictions

The capture function must be running.

Format

```
►►──ZFCAP ALTER── ──ECB── ──LEVEL── ──max──────────────────────────────────►◄
```

max
is a decimal number from 1–32 that specifies the maximum number of captures allowed at one time.

Additional Information

- Enter the ZFCAP STATUS command to display the current maximum number of captures that are allowed at one time.
- The ZFCAP CLEAR and ZFRST CLEAR commands reset the maximum number of captures to the default value defined in the master copy of the capture and restore working keypoint.

Examples

The maximum number of captures allowed at one time is changed in the following example.

```
User:   ZFCAP ALTER ECB LEVEL 16

System: FCAP0084I 10.57.11 ECB LEVEL ALTERED TO 16
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP CHANGE–Change the Capture Maximums

Use this command to change the maximum number of captures allowed for each tape control unit, tape channel path, and DASD channel path.

Requirements and Restrictions

- You can change the capture maximums for a processor only if the capture and restore working keypoint for that processor is initialized. Otherwise, an OPR-041500 dump occurs.
- The maximum number of captures for each DASD control unit is always 1. You cannot change this value.

Format

```
►►──ZFCAP CHAnge──┬────────────┬─  ──┬─TAPECU-─┬──max──────────────────────►◄
                  ├─ ──CPU-id──┤     ├─DASDCU-─┤
                  └─ ──ALL─────┘     ├─DASDCH-─┤
                                     └─TAPECH-─┘
```

CPU-*id*
: changes the capture maximums for a specific processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

ALL
: changes the capture maximums for all the processors.

TAPECU
: specifies the maximum number of captures allowed for each tape control unit.

DASDCU
: specifies the maximum number of captures allowed for each direct access storage device (DASD) control unit.

DASDCH
: specifies the maximum number of captures allowed for each DASD channel path.

TAPECH
: specifies the maximum number of captures allowed for each tape channel path.

max
: is a decimal number from 1–255 that specifies the new capture maximum.

Additional Information

- You can change the capture maximums for both active and inactive processors.
- If you do not specify the CPU or ALL parameter, the capture maximums are changed for the current processor.
- The CPU and ALL parameters are ignored if the system is not generated for loosely coupled support.
- Enter the ZFCAP DISPLAY command to display the current capture maximums.
- For DASD subsystems that are identified by subsystem ID (SSID), a DASD control unit represents all devices that are associated with that SSID.

- The ZFCAP CLEAR and ZFRST CLEAR commands reset the capture maximums to the default values defined in the master copy of the capture and restore working keypoint.

Examples

The maximum number of captures allowed for each tape control unit is changed for the current processor in the following example.

```
User:   ZFCAP CHANGE TAPECU-3

System: FCAP0155I 10.46.45 COMPLETED
```

The capture maximums are changed for all the processors in the following example.

```
User:   ZFCAP CHANGE ALL TAPECU-3 TAPECH-2 DASDCH-2

System: FCAP0155I 08.48.11 COMPLETED
```

Related Information

See *TPF Database Reference* for information about determining the optimal values for the capture maximums.

ZFCAP CLEAR–Initialize the Capture and Restore Working Keypoint

Use this command to initialize the capture and restore working keypoint. The fields in the keypoint are reset to the default values stored in the master copy of the keypoint.

Requirements and Restrictions

If you enter the ZFCAP CLEAR command with the BP parameter while the capture function is running, you must enter the ZTDEV RESUME command for each tape device that was previously included in the capture and restore working keypoint.

Format

```
►►──ZFCAP CLEAR──┬─────────┬──────────────────────────────────────────►◄
                 ├──ALL──┤
                 └──BP───┘
```

ALL
initializes the capture and restore working keypoint on all active processors if the capture function is not running. If you do not specify the ALL parameter, the capture and restore working keypoint is initialized only on the current processor.

BP
initializes the capture and restore working keypoint on the current processor when the capture function is running.

Attention: Initializing the capture and restore working keypoint when the capture function is running causes unexpected results. Use this parameter only when other attempts to end the capture function fail.

Additional Information

- This command resets the following values:
  - Capture maximums that were defined using the ZFCAP CHANGE command
  - Maximum number of captures allowed at one time that were defined using the ZFCAP ALTER command
  - Input/output (I/O) delay time factor that was defined using the ZFCAP IOTIME command.
  - Maximum number of restores allowed at one time that were defined using the ZFRST ALTER command.
- You can also enter the ZFRST CLEAR command to initialize the capture and restore working keypoint.

Examples

The capture and restore working keypoint is initialized on the current processor in the following example.

```
User:   ZFCAP CLEAR

System: FCAP0060I 08.52.29 FCAP/FRST KEYPOINT REINITIALIZED
```

Related Information

See *TPF Database Reference* for more information about the capture and restore working keypoint.

ZFCAP DISPLAY–Display the Capture Maximums

Use this command to display the maximum number of captures allowed for each tape control unit, tape channel path, and DASD channel path.

Requirements and Restrictions

You can display the capture maximums for a processor only if the capture and restore working keypoint for that processor is initialized. Otherwise, an OPR-041500 dump occurs.

Format

```
►►──ZFCAP DISplay──┬──────────┬──────────────────────────────────►◄
                   ├─ ──CPU-id─┤
                   └─ ──ALL────┘
```

CPU-*id*
: displays the capture maximums for a specific processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

ALL
: displays the capture maximums for all the processors.

Additional Information

- If you do not specify the CPU or ALL parameter, the capture maximums are displayed for the current processor.
- The CPU and ALL parameters are ignored if the system is not generated for loosely coupled support.
- Enter the ZFCAP CHANGE command to change the capture maximums.

Examples

Information about the capture maximums for CPU D is displayed in the following example.

```
User:   ZFCAP DISPLAY CPU-D

System: FCAP0153I 08.45.35 FOR CPU-D  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
```

Information about the capture maximums for all the processors is displayed in the following example.

```
User:   ZFCAP DISPLAY ALL

System: FCAP0153I 08.45.38 FOR CPU-B  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0153I 08.45.38 FOR CPU-C  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0153I 08.45.38 FOR CPU-D  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0153I 08.45.38 FOR CPU-E  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0153I 08.45.38 FOR CPU-Z  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0153I 08.45.38 FOR CPU-0  DASDCU-1 DASDCH-1 TAPECH-1 TAPECU-1
        FCAP0155I 08.45.38 COMPLETED
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP IOTIME–Display or Alter I/O Delay Time Factor

Use this command to display or change the value of the input/output (I/O) delay time factor (BXIOTIME) in the capture and restore working keypoint on the current processor.

Requirements and Restrictions

None.

Format

```
►►──ZFCAP IOTIME──┬─────────┬──────────────────────────────────────►◄
                  └──delay──┘
```

delay
: is a decimal number from 1–9999 that specifies the new I/O delay time factor. This value is a multiple of 1/256th of a second. If you omit this parameter, the current I/O delay time factor is displayed.

Additional Information

When you determine the optimal value for the I/O delay time factor, update the master copy of the capture and restore working keypoint as well. This ensures that the value is retained if the keypoint is initialized using the ZFCAP CLEAR or ZFRST CLEAR command.

Examples

The current I/O delay time factor is displayed in the following example.

```
User:   ZFCAP IOTIME

System: FCAP0090I 10.53.55 CURRENT VALUE OF I/O DELAY TIME FACTOR IS 0004
```

The I/O time delay factor is changed in the following example.

```
User:   ZFCAP IOTIME 0005

System: FCAP0140E 08.54.07 VALUE OF I/O DELAY TIME FACTOR ALTERED TO 0005
```

Related Information

See *TPF Database Reference* for more information about the I/O delay time factor.

ZFCAP LOG–Start or Stop Record Logging

Use this command to start or stop record logging for all active processors.

Requirements and Restrictions

- Use the ZTMNT command to mount the logging tapes (RTT–RTY) on all the active processors.
- Do not use the ALL prefix with this command.

Format

```
►►──ZFCAP LOG──  ──┬─START─┬──────────────────────────────────────────────►◄
                   └─STOP──┘
```

START
starts record logging for all the active processors.

STOP
stops record logging for all the active processors.

Additional Information

Use the ZFRST LOG command to restore the logging records.

Examples

Record logging is started for all active processors in the following example.

```
User:   ZFCAP LOG START

System: FCAP0023I 10.50.18 LOGGING STARTED - PROCESSOR B
```

Record logging is stopped for all active processors in the following example.

```
User:   ZFCAP LOG STOP

System: FCAP0024I 14.02.53 LOGGING STOPPED - PROCESSOR B
        FCAP0024I 14.02.53 LOGGING STOPPED - PROCESSOR C
        COTG0080A 14.02.53 TOFF BSS    REMOVE RTU FROM DEVICE 422
                               VSN A00065 G0002 S0001 D38K   SL  NOBLK  NOCOMP
        COTG0080A 14.02.53 TOFF BSS    REMOVE RTU FROM DEVICE 522
                               VSN A00132 G0007 S0001 D38K   SL  NOBLK  NOCOMP
```

Related Information

See *TPF Database Reference* for more information about record logging.

ZFCAP MISC–Start the Keypoint Capture Function

Use this command to capture the system keypoint records to the keypoint capture (KPC) tape or to restart the keypoint capture function after an initial program load (IPL) is performed on the system.

Requirements and Restrictions

- Use the ZTMNT command to mount the KPC tape.
- The processor must be in UTIL state or higher.
- Keypoints that are shared between subsystems are captured only when the keypoint capture function is started for the basic subsystem (BSS).
- Configuration-dependent keypoints (for example, keypoint 1, keypoint 6, and keypoint I) are not captured.
- Do not use the ALL prefix with this command.

Format

```
►►──ZFCAP MISC── ──┬─S─┬── ──KPT──────────────────────────────────────────►◄
                   └─R─┘
```

S starts the keypoint capture function.

R restarts the keypoint capture function after an IPL.

Additional Information

- The keypoint capture function is automatically started when you enter the ZFCAP SEXPT command to end exception recording after the capture function is completed for all the online modules (ZFCAP ALL). (It is not automatically started if only specific modules were captured.)
- Use the ZFRST KPT command to restore the system keypoint records.

Examples

The system keypoint records are captured in the following example.

```
User:   ZFCAP MISC S KPT

System: FCAP0025I 13.57.48 KPT STARTED
        FCAP0046I 13.57.52 KEYPOINT I NOT CAPTURED
        FCAP0046I 13.57.52 KEYPOINT 1 NOT CAPTURED
        FCAP0046I 13.57.52 KEYPOINT 1 NOT CAPTURED
        FCAP0046I 13.57.52 KEYPOINT 6 NOT CAPTURED
        FCAP0027I 13.57.54 KPT COMPLETE
        COTC0080A 13.57.54 TCLS HPN    REMOVE KPC FROM DEVICE 425
                               VSN A00068 G0003 S0001 D38K   SL  NOBLK  NOCOMP
```

The keypoint capture function is started again in the following example.

```
User:   ZFCAP MISC R KPT

System: FCAP0026I 14.01.21 KPT RESTARTED
        FCAP0046I 14.01.25 KEYPOINT I NOT CAPTURED
        FCAP0046I 14.01.25 KEYPOINT 1 NOT CAPTURED
        FCAP0046I 14.01.25 KEYPOINT 1 NOT CAPTURED
        FCAP0046I 14.01.25 KEYPOINT 6 NOT CAPTURED
        FCAP0027I 14.01.26 KPT COMPLETE
        COTC0080A 14.01.26 TCLS HPN    REMOVE KPC FROM DEVICE 425
                               VSN A00068 G0004 S0001 D38K   SL  NOBLK  NOCOMP
```

Related Information

See *TPF Database Reference* for more information about the keypoint capture function.

ZFCAP PAUSE–Pause the Capture Function

Use this command to temporarily stop (pause) the capture function for the current processor or for all the participating processors. If the capture function is capturing a module (physical DASD unit) when you enter this command, the capture for that module is completed before the function is paused.

Requirements and Restrictions

- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this function news up; use the ALL parameter instead.

Format

```
►►──ZFCAP PAUSE──┬───────┬──────────────────────────────────────────►◄
                 └─ALL───┘
```

ALL
temporarily stops the capture function for all the participating processors. If you do not specify the ALL parameter, the capture function is paused only for the current processor.

Additional Information

Exception recording and record logging continue if they were running.

Examples

The capture function is paused for the current processor in the following example.

```
User:   ZFCAP PAUSE

System: FCAP0043I 10.18.50 PAUSE IN PROGRESS PROCESSOR B
        COTM0046I 10.22.43 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 424
                           VSN A00245 G     S     D38K   SL  NOBLK  NOCOMP
        COTS0382I 10.22.44 TPSW WP1    TAPE BXA SWITCHED FROM 423 TO 424
                           VSN IS NOW A00245
        COTS0080A 10.22.44 TWEV WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK  NOCOMP
        COTM0046I 10.22.48 TMNT WP1    TAPE BXB MOUNTED ON DEVICE 481
                           VSN A00249 G     S     D38K   SL  NOBLK  NOCOMP
        COTS0382I 10.22.48 TPSW WP1    TAPE BXB SWITCHED FROM 480 TO 481
                           VSN IS NOW A00249
        COTS0080A 10.22.48 TWEV WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248 G     S     D38K   SL  NOBLK  NOCOMP
        COTC0080A 10.23.56 TCLS WP1    REMOVE BXA FROM DEVICE 424
                           VSN A00245 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0009I 10.23.56 MOD 01E ON 03E7 TO TAPE ON 424 - COMPLETED
        FCAP0009I 10.23.59 MOD 01F ON 0EEB TO TAPE ON 481 - COMPLETED
        FCAP0018I 10.23.59 PROCESSOR B PAUSED
        COTC0080A 10.23.59 TCLS WP1    REMOVE BXB FROM DEVICE 481
                           VSN A00249 G     S     D38K   SL  NOBLK  NOCOMP
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP RESTART–Restart the Capture Function

Use this command to start the capture function again after you temporarily stop (pause) it using the ZFCAP PAUSE command. You can use this command to restart the capture function after an initial program load (IPL) is performed for the TPF system.

Requirements and Restrictions

- Load the capture tapes (BXA–BXZ, BX0–BX5), initialize them, and ensure that they are ready to use.
- Each participating processor must be in UTIL state or higher.

Format

```
►►—ZFCAP RESTART—┬───────┬——————————————————————————————————————►◄
                 └—ALL———┘
```

ALL
starts the capture function again for all the participating processors. If you do not specify the ALL parameter, the capture function is started again only for the current processor.

Additional Information

Use the ZFRST CAP command to restore the online modules (physical DASD units).

Examples

The capture function is started again for the current processor in the following example.

```
User:   ZFCAP RESTART

System: FCAP0014I 10.31.40 PROCESSOR B RESTART COMPLETE
        FCAP0008I 10.31.40 MOD 01F ON 0EEB TO TAPE ON 424 - STARTED
        COTM0046I 10.31.40 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 424
                           VSN A00245 G     S     D38K   SL  NOBLK  NOCOMP
        COTM0046I 10.33.24 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK  NOCOMP
        COTS0382I 10.33.24 TPSW WP1    TAPE BXA SWITCHED FROM 424 TO 423
                           VSN IS NOW A00243
        COTS0080A 10.33.24 TWEV WP1    REMOVE BXA FROM DEVICE 424
                           VSN A00245 G     S     D38K   SL  NOBLK  NOCOMP
        COTC0080A 10.34.06 TCLS WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK  NOCOMP
        FCAP0009I 10.34.06 MOD 01F ON 0EEB TO TAPE ON 423 - COMPLETED
        FCAP0001I 10.34.06
        CAPTURE STARTED 06MAY 1027
        MODULES TO DO        2
        MODULES COMPLETED    2
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE   Y
        IOTIME                004
        DISK ERRORS
        TRKS 000
        RCDS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   423   424   C0              B
        END OF STATUS DISPLAY
        FCAP0134I 10.34.07 PROCESSOR B COMPLETED
        FCAP0011I 10.34.07 COMPLETED
        FCAP0044I 10.34.07 EXCEPTION RECORDING CONTINUING
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP STATUS–Display Capture Status

Use this command to display the status and progress of the capture function.

Requirements and Restrictions

None.

Format

```
►►──ZFCAP STATUS──┬─────────────────────────────────┬──────────────────────►◄
                  └──MODULE──  ─┬─module─────┬─┘
                                ├─ALL────────┤
                                ├─COMPLETED──┤
                                ├─INPROGRESS─┤
                                ├─REMAINING──┤
                                ├─ABORTED────┤
                                └─DEFERRED───┘
```

MODULE
: displays information that indicates whether the capture function completed, is in progress, was not started yet, was stopped, or was deferred for a specific module (physical DASD unit). If you do not specify the MODULE parameter, overall status information about the capture function is displayed.

module
: is a 3-digit hexadecimal symbolic module number. When specified, information is displayed that indicates whether the capture function completed, is in progress, was not started yet, was stopped, or was deferred for that module.

ALL
: lists all the modules and indicates, for each module, whether the capture function completed, is in progress, was not started yet, was stopped, or was deferred.

COMPLETED
: lists the modules where the capture function completed.

INPROGRESS
: lists the modules where the capture function is in progress.

REMAINING
: lists the modules where the capture function was not started yet.

ABORTED
: lists the modules where the capture function was stopped

DEFERRED
: lists the modules where the capture function was deferred.

Additional Information

None.

Examples

Overall status information about the capture function is displayed in the following example.

```
User:   ZFCAP STATUS

System: FCAP0001I 10.42.10
        CAPTURE STARTED 06MAY 1041
        MODULES TO DO        2
        MODULES COMPLETED    0
        MODULES REMAINING    2
                              PROCESSOR
                              B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    002
        XCP RECORDING ACTIVE   Y
        IOTIME                004
        DISK ERRORS
        TRKS 000
        RCDS 000

            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   423   424   C0   / 00 05     B
          BXB   480   481   C0   / 00 05     B
        INPROGRESS
        MOD 01E ON DEV 03E7  TO  TAPE 423 STARTED 1041
        MOD 01F ON DEV 0EEB  TO  TAPE 480 STARTED 1042
        END OF STATUS DISPLAY
```

Status information about the capture function for module 01E is displayed in the following example.

```
User:   ZFCAP STATUS MODULE 01E

System: FCAP0126I 15.39.22  MOD 01E IS INPROGRESS
```

Status information about all the modules is displayed in the following example.

```
User:   ZFCAP STATUS MODULE ALL

System: FCAP0163I 15.40.10 CAPTURE MODULE STATUS
        FCAP - MODULES COMPLETED
                NO MODULES FOUND
        FCAP - MODULES INPROGRESS
            01E 01F
        FCAP - MODULES REMAINING
                NO MODULES FOUND
        FCAP - MODULES ABORTED
                NO MODULES FOUND
        FCAP - MODULES DEFERRED
                NO MODULES FOUND
        DISPLAY COMPLETE
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFCAP TAPE–Add or Delete Tape Devices

Use this command to make additional primary and alternate tape devices available to the capture function or to delete primary tape devices (and the corresponding alternate tape devices, if they exist). After a tape device is deleted, it is no longer available to the capture function.

Requirements and Restrictions

- Enter this command from the processor where you want to add or delete the tape device.
- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format

```
                         ┌─/──────────────────────┐
                         ▼                        │
►►──ZFCAP TAPE── ──┬─ADD── ───tapeaddr─┬─────────┬─┴──┬──────────────────►◄
                   │                   └─-altaddr─┘    │
                   ├─ALT── ──tapeaddr-altaddr──────────┤
                   │            ┌─/─────────┐          │
                   │            ▼           │          │
                   └─DEL── ──┬───tapeaddr─┴─┬─────────┘
                             └─ALL── ──cu───┘
```

ADD
: adds primary tape devices and corresponding alternate tape devices.

ALT
: adds alternate tape devices for existing primary tape devices.

DEL
: deletes primary tape devices and the corresponding alternate tape devices, if they exist.

tapeaddr
: is the 3-digit hexadecimal address of a primary tape device.

altaddr
: is the 3-character hexadecimal address of an alternate tape device.

Note: The addresses of the primary and alternate tape devices must be on the same device type.

ALL *cu*
: deletes all the primary and alternate tape devices on the specified control unit, where *cu* is the 2-digit hexadecimal address of the control unit.

Additional Information

If you delete a tape device while it is being used to capture a module, the capture function temporarily ends for that module and then automatically starts again later.

Examples

Two primary tape devices are added for the capture function in the following example.

```
User:   ZFCAP TAPE ADD 424/425

System: FCAP0021I 13.37.55 TAPE 424 ADDED
        FCAP0021I 13.37.55 TAPE 425 ADDED
```

A primary tape device and an alternate tape device are added for the capture function in the following example.

```
User:   ZFCAP TAPE ADD 480-481

System: FCAP0119I 10.36.50 DUAL TAPE 480-481 ADDED
```

An alternate tape device is added for tape device 423 in the following example.

```
User:   ZFCAP TAPE ALT 423-424

System: FCAP0119I 13.39.29 DUAL TAPE 423-424 ADDED
```

An alternate tape device is added for tape devices 423 and 480 in the following example.

```
User:   ZFCAP TAPE ALT 423-424/480-481

System: FCAP0119I 10.39.33 DUAL TAPE 423-424 ADDED
        FCAP0119I 10.39.33 DUAL TAPE 480-481 ADDED
```

Two primary tape devices and any corresponding alternate tape devices are deleted in the following example. These tape devices are no longer available to the capture function.

```
User:   ZFCAP TAPE DEL 424/425

System: FCAP0019I 13.38.45 TAPE 424 DELETED
        FCAP0019I 13.38.45 TAPE 425 DELETED
```

Related Information

See *TPF Database Reference* for more information about the capture function.

ZFDNT–Dismount General File

Use this command to make a general file data set unavailable to the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZFDNT── ──number──────────────────────────────────────────────────►◄
```

number
is a data set number from 0–99.

Additional Information

None.

Examples

The specified data set is dismounted in the following example.

```
User:   ZFDNT 03

System: FDNT0001I 09.02.14 GENERAL FILE DATA SET  3 DISMOUNTED FROM 04E7
```

Related Information

See *TPF Database Reference* for more information about general file support.

ZFECB–Display Active ECB Information

Use this command to display a list of active entry control blocks (ECBs) or a formatted display of a particular ECB and any data event control blocks (DECBs) attached to that ECB.

Requirements and Restrictions

You can only enter this command in a basic subsystem (BSS), although it can be used to display information about non-BSS ECBs.

Format

```
►►──ZFECB──┬──────────────────────────┬──────────────────────────────────►◄
           ├──time──┬──────────────┬──┤
           │        ├─/M───────────┤  │
           │        ├─/H───────────┤  │
           │        └─/svcname─────┘  │
           ├──address─────────────────┤
           └──*lniata─────────────────┘
```

time
: is the number of seconds or minutes for which you want to check active ECBs. Specify a 3 digit number from 000–999. If you do not specify any additional parameters, a list of ECBs that are older than the specified time, in seconds, is displayed.

M
: displays a list of ECBs that are older than the input time in minutes.

H
: displays a list of ECBs that are older than the input time, in seconds, that are also holding one or more records.

svcname
: is a supervisor call (SVC) mnemonic. This parameter displays a list of ECBs that are older than the input time, in seconds, and whose last SVC issued was the SVC entered.

address
: is a valid 6- or 8-character ECB system virtual memory (SVM) address. This parameter displays a formatted version of the specified ECB.

******lniata*
: displays a list of active ECBs, where *lniata* is the terminal address.

Additional Information

Enter **ZFECB** without any parameters to display the state change ECB.

Examples

The following example shows an active ECB display, where:

ECB
: is the SVM address of the active ECB.

SVC
: is the primary SVC last issued by the ECB.

PROG
: is the name of the program that issued the last SVC.

TIME
: is the amount of time, in minutes (0000) and seconds (00), that this ECB has been active.

TYPE-ECB
: is the method in which the ECB was created. If it was created with a create macro, the display also shows the name of the creating program.

IO
: indicates if input/output (IO) is outstanding for this ECB.

BLK
: is the number of core blocks in use by this ECB.

HLD
: is the number of records held by this ECB.

PSW
: is the program status word (PSW) as last saved in field CE1PSW of the ECB.

IS
: is the I-stream number that is currently assigned.

ACTIVATION NUMBER
: is the activation number of the program currently in use by this ECB.

```
User:   ZFECB 000

System: FECB0006I 13.04.54 ACTIVE ECB DISPLAY

          ECB  SVC PROG      TIME   TYPE-ECB IO BLK HLD        PSW
        IS-1  ACTIVATION NUMBER- 1
        BF9000 0C UDE0     0000 27 1052      X  003  01 071D0000821C90D6
        IS-1  ACTIVATION NUMBER- 1
        C29000 0C UDE0     0000 33 1052      X  002  00 071D0000821C90D6
        IS-1  ACTIVATION NUMBER- 1
        C44000 0A CVCJ     0000 17 1052      X  001  00 070D00008361456E
        IS-1  ACTIVATION NUMBER- 1
        C47000 0C UDE0     0000 30 1052      X  002  00 071D0000821C90D6
        IS-1  ACTIVATION NUMBER- 1
        C4A000 0A CVCX     0000 17 CREM/CVCJ X  000  00 071D300082288028
        IS-1  ACTIVATION NUMBER- 1
        D07000 F8 CSGM     0000 10 CREM/CSGA X  000  00 070D3000836073EA
        IS-1  ACTIVATION NUMBER- 1
        D13000 0A CAPN     0000 10 CREM/CTKT X  000  00 071D1000835ECAFE
        IS-1  ACTIVATION NUMBER- 1
        E0F000 A0 CDE3     0000 00 1052         003  00 071D0000821CA538
        TOTAL ECB 008  HCB
        END OF DISPLAY
```

The following example shows a formatted ECB display, which includes the following information:

- The active program line of the display shows as many as five nested program names.
- The D0–DF labels represent data levels 0–F in the ECB. The first 4 bytes contain the file address and the second 4 bytes contain the SVM translation of the core block address.
- The first 4 bytes of the FMP label contain the data in field CE1FMP of the ECB. The second 4 bytes contain the SVM translation of the data in field CE1CRP of the ECB.
- Labels R0–R8 contain the translated SVM addresses (if applicable) of the values contained in service call register save areas CE1SVR–CE1SVP.

- Labels R14–R15 contain the translated SVM addresses (if applicable) of the values contained in service call register save areas CE1RDA and CE1RDB.
- Label R9 contains the translated ECB SVM address.
- Labels URA–UR7 contain the translated SVM addresses (if applicable) of the values contained in user register save areas CE1URA–CE1UR7.
- Label DECB contains the SVM of the DECB.
- Label NAME is the name of the DECB.
- Label FARW indicates the file address reference word (FARW) for the DECB indicated. The first 4 bytes contain the record ID and record code check (RCC) for this FARW, and the last 8 bytes contain the file address.
- Label CBRW contains the core block reference word (CBRW) for the DECB indicated. The first 4 bytes contain the SVM translation of the core block address, the next 2 bytes indicate the block type, and the last 2 bytes indicate the block size.
- Label SUD is the error indicator for this DECB.

```
User:   ZFECB DDF000

System: FECB0008I 13.08.33 FORMATTED ECB DISPLAY

        ACTIVE PROGRAM LOADSET -  BASE
        ACTIVE PROG CVCX 028                                    IS-1
        R0  00000000 D0    D006C007 0052E000 D8    D006C006 0052E000
        R1  00000000 D1    D006C001 0052E000 D9    00000000 00000000
        R2  00173B60 D2    CC01C801 005DBC00 DA    00000000 00000000
        R3  0053E000 D3    CC01E800 005DBC00 DB    00000000 00000000
        R4  0017A000 D4    CC01F871 005DBC00 DC    00000000 00000000
        R5  00173000 D5    1801D844 005E9000 DD    00000000 00000000
        R6  00000052 D6    CC003837 005DCBE0 DE    CC03E003 0052C000
        R7  00000018 D7    D006C006 005E9000 DF    00000000 005DBE80
        R8  02288000 FMP   D4038701 02288000 PSW   071D3000 82288028
        R9  00C4A000 EBSW  00000000 09000000 CTL   00010000 39000000
        R14 0051BC00 EBXS  08000003 00000000 REC   00010000 00000000
        R15 00000004 SUD   00000000 00000000 00000000 00000000 00 00
        W000 00000000 00000000  X000 C3E3D2E3 00000001  URA 822741C4
        W008 00000000 00000000  X008 0000000C 00000000  URB 82274224
        W016 C3C3D4D3 F3C40000  X016 00000004 00000000  UR0 00000004
        W024 00000000 00000C3E  X024 00000000 00000000  UR1 0000070C
        W032 0019FCD8 0019FDA0  X032 00000000 00000000  UR2 0101F454
        W040 0019FCA8 00000000  X040 00000000 00000000  UR3 0052F000
        W048 00000000 00000000  X048 00000000 00000000  UR4 0051BC00
        W056 00000000 00000000  X056 00000000 00000000  UR5 0017A000
        W064 00000000 00000000  X064 00000000 00000000  UR6 00502010
        W040 0019FCA8 00000000  X040 00000000 00000000  UR3 0052F000
        W048 00000000 00000000  X048 00000000 00000000  UR4 0051BC00
        W056 00000000 00000000  X056 00000000 00000000  UR5 0017A000
        W064 00000000 00000000  X064 00000000 00000000  UR6 00502010
        W072 00000000 04000000  X072 00000000 00000000  UR7 8053100E
        W080 02000000 E7E20000  X080 00000000 00000000
        W088 001EAE00 00000000  X088 00000000 00000000
        W096 00000050 00000000  X096 00000000 00000000
        PRL 005022D0   PBI FF00         SUC 0010
        TRC C3E5C3D1   DBI FF00         HLD 00
        PAT 03F260C0   SSU FF00         PID C2
        CHW 00000000
        BAD 80115A56
        FA0 00000000   CT0 00010000     FA8 C8D90000   CT8 00010FFF
        FA1 00000000   CT1 00010000     FA9 00000000   CT9 00010000
        FA2 00000000   CT2 00010000     FAA 00000000   CTA 00010000
        FA3 00000000   CT3 00010000     FAB 00000000   CTB 00010000
        FA4 00000000   CT4 00010000     FAC 00000000   CTC 00010000
        FA5 00000000   CT5 00010000     FAD 00000000   CTD 00010000
        FA6 00000000   CT6 00010000     FAE 00000000   CTE 00010000
        FA7 00000000   CT7 00010000     FAF 00000000   CTF 00010000

        DECB 0053B008 NAME D7D5D900 00000000 00000000 00000000
          FARW 00000000 D006C005 C8D90000 CBRW 0052F000 00010FFF SUD 00
        END OF DISPLAY
```

The following example displays a list of all active ECBs whose last SVC issued was the DLAYC macro.

```
User:   ZFECB 000/DLAYC

System: FECB0006I 11.46.08 ACTIVE ECB DISPLAY
          ECB  SVC PROG      TIME   TYPE-ECB IO BLK HLD        PSW
        IS-1  ACTIVATION NUMBER- 4
        2E4A000 0C UDE0     0000 51 1052      X  002  00 071D0000857810D6
        IS-1  ACTIVATION NUMBER- 4
        2E56000 0C UDE0     0000 46 1052      X  003  01 071D0000857810D6
        TOTAL ECB 002  HCC
        END OF DISPLAY
```

Related Information

- See *TPF Concepts and Structures* for more information about ECBs and DECBs.
- See *TPF System Installation Support Reference* for more information about user exit UDE0, which can be used to tailor the list of active ECBs.
- See *TPF Application Programming* for more information about DECBs.

ZFILE–Activate a TPF Segment or Script

Use this command to activate a TPF segment or script from the command line.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- The activated segment must contain a `main` function.

Format

```
                         ┌──────────────┐ ┌──────────────────┐
                         ▼              │ ▼                  │
►►──ZFILE── ──path───────┴──────────────┴─┴──────────────────┴───────────────►◄
                           └── ──-option─┘   └── ──parameter─┘
```

path
: is the path of the file, which contains the call information for the segment or script to be activated.
  - If you are activating a TPF segment, the first line of the file must contain the name of the segment to be activated in the form #!*xxxx*, where *xxxx* is the TPF segment name.
  - If you are activating a script, the first line of the file must contain the file system file used to process the script in the form #!/*pppp*/*ffff*, where *pppp* is the path to the file system file and *ffff* is the name of that file.

option
: is an option that you code into the segment or script being activated. The options, which always begin with a dash (-), are based on how the segment or script was coded. If you specify more than one option, you can specify these options separately or together. For example, if -l and -a are valid options, you can enter one of the following:
  - **ZFILE** *path* **-l -a** *parameter*
  - **ZFILE** *path* **-la** *parameter*

parameter
: is an input to *path*, which you code in the segment or script being activated. The parameters are based on how the segment or script was coded.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- Before you activate a script or TPF segment, set the access permissions of the file to execute (or run). See "ZFILE chmod–Change the Access Permissions of a File or Directory" on page 444 for more information about setting access permissions.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, a TPF segment is activated from the command line. File `/usr/local/test` contains the string `#!QZZ8` in the first line of the file. For this, TPF segment QZZ8 is written in C language and contains a function to print `Hello, World!`

```
User:   ZFILE /usr/local/test

System: FILE0001I 08:14:31 START OF DISPLAY FROM /usr/local/test

        Hello, World!

        END OF DISPLAY
```

In the following example, a script stored in a file is activated from the command line. The current working directory is `/usr/bin`. File `perl`, representing the *path* parameter, contains the string `#!QZZ9` in the first line of the file. For this, TPF segment QZZ9 contains the TPF code for a scripting language such as the Perl interpreter. File `script`, representing the *parameter* parameter, contains the following two lines:

```
#!/usr/bin/perl
print 'Hello, World!'
```

The scripting language is called through file `perl` and is passed the parameter `script`. The scripting language interprets the input and the result is displayed.

```
User:   ZFILE script

System: FILE0001I 08:14:31 START OF DISPLAY FROM script

        Hello, World!

        END OF DISPLAY
```

In the following example, a script is entered on the command line. The file `/usr/bin/perl` contains the string `#!QZZ9` in the first line of the file. For this, TPF segment QZZ9 contains the TPF code for a scripting language such as the Perl interpreter. The `-e` option and `'print "Hello, World!\n";'` are passed to the scripting language as an option and a parameter, respectively. The scripting language interprets the input and the result is displayed.

```
User:   ZFILE /usr/bin/perl -e 'print "Hello, World!\n";'

System: FILE0001I 08:14:31 START OF DISPLAY FROM perl -e 'print "Hello, Wor...

        Hello, World!

        END OF DISPLAY
```

Related Information

- “ZFILE chmod–Change the Access Permissions of a File or Directory” on page 444
- See *TPF Application Programming* for more information about coding a C language `main` function.

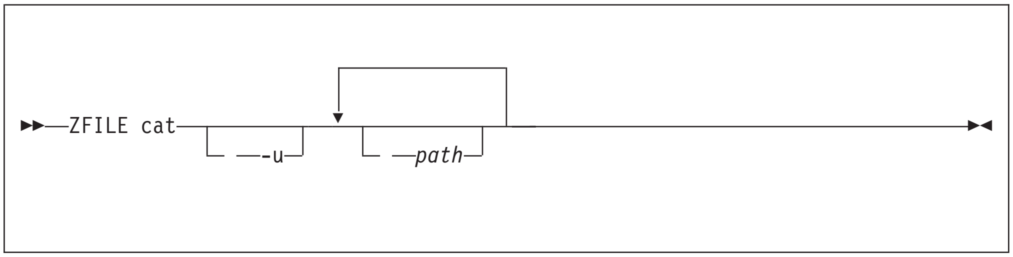
ZFILE cat–Concatenate and Write Files

Use this command to concatenate files and write the contents to the standard output (`stdout`) stream.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



-u writes each byte without a delay; that is, it does not use buffered input/output (I/O).

path
is the path name of the file whose contents are to be concatenated and written to the standard output (`stdout`) stream.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP cat
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE cat command is based on a subset of the cat utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	**Description**
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the contents of `test.file` and `next.file` are concatenated and displayed.

```
User:   ZFILE cat test.file next.file

System: FILE0001I 13:11:31 START OF DISPLAY FROM cat test.file next.file
        abc
        def
        END OF DISPLAY
```

In the following example, the contents of `my.file` and `your.file` are concatenated and written to a new file called `our.file`.

```
User:   ZFILE cat my.file your.file > our.file

System: FILE0003I 14:05:55 cat my.file... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE cd–Change the Current Working Directory

Use this command to change the current working directory of your TPF terminal.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE cd──┬──/─────┬──────────────────────────────────────►◄
              ├──path──┤
              └──-─────┘
```

path
: is the path name of the directory to which you are changing. If you do not specify this parameter, the default directory, which is initially set to the root directory (/), is used. You can also use one of the following in place of a specific directory:

  . leaves you in the current directory.

  .. changes the current directory to the parent of the current directory.

- changes the current directory to the previous directory.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP cd
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE cd command is based on a subset of the cd utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‛), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the current directory is changed to /tmp.

```
User:   ZFILE cd /tmp

System: FILE0003I 15:31:22 cd /tmp COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE pwd–Write the Current Working Directory” on page 516 for information about how to find the current working directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE chmod–Change the Access Permissions of a File or Directory

Use this command to change the *access permissions* of a file or directory. The access permissions determine who can read, write, or search a file or directory.

Note: In a program, you can set the access permissions of a file or directory with the `chmod` function.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE chmod──┬──────┬── ──permission── ──path──────────────────────►◄
                 └──-R──┘
```

-R recursively changes the access permissions of the specified file or directory. If you specify a directory, the access permissions of all files and subdirectories under that directory are changed. If you do not specify this parameter, the access permissions for only the specified file or directory are changed.

permission
is the permission setting specified in octal notation. Specify this permission setting by adding the following values:

4000
sets the effective user ID (UID) of the process that executes the file. In the `chmod` function, this value is S_ISUID. This setting has no effect on directories.

2000
sets the effective group ID (GID) of the process that executes the file. In the `chmod` function, this value is S_ISGID. If you set this value for a directory, all files created in that directory will have the same GID of that directory.

0400
allows the owner to read the file or directory. In the `chmod` function, this value is S_IRUSR.

0200
allows the owner to write to the file or directory. In the `chmod` function, this value is S_IWUSR.

0100
allows the owner to execute (or run) the file or search the directory. In the `chmod` function, this value is S_IXUSR. Specify this permission only for directories or regular files; you cannot set this permission for special files.

0040
allows the group to read the file or directory. In the `chmod` function, this value is S_IRGRP.

0020
allows the group to write to the file or directory. In the `chmod` function, this value is S_IWGRP.

0010
allows the group to execute (or run) the file or search the directory. In the `chmod` function, this value is S_IXGRP. Specify this permission only for directories or regular files; you cannot set this permission for special files.

0004
allows others to read the file or directory. In the `chmod` function, this value is S_IROTH.

0002
allows others to write to the file or directory. In the `chmod` function, this value is S_IWOTH.

0001
allows others to execute (or run) the file or search the directory. In the `chmod` function, this value is S_IXOTH. Specify this permission only for directories or regular files; you cannot set this permission for special files.

For example, to set the access permissions to allow the owner to read, write, and execute the file, and to allow the group and others to read and execute the file, specify 755 for *permission*.

Note: You do not have to specify the leading zeros for the access permission values.

path
is the path name of the file or directory whose access permissions you are changing.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP chmod
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You must set on the execute permission to use a directory as a directory; for example, to list the directory contents or to remove files.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.

- The ZFILE chmod command is based on a subset of the chmod utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

  Wildcard — **Description**

  ***** — Matches any string, including the empty string.

  ? — Matches any single character.

  [..] — Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )

number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the access permissions for `/tmp/sample.1` are set to read and write by owner and group.

```
User:   ZFILE chmod 0660 /tmp/sample.1

System: FILE0003I 15.26.53 chmod 0660 ... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE ls–List File and Directory Names and Attributes” on page 491 for information about how to display the current access permissions of a file or directory.
- See “ZFILE chown–Change the Owner and Group of a File or Directory” on page 448 for information about how to change the owner or group of a file or directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and the *TPF C/C++ Language Support User’s Guide* for more information about user IDs (UIDs), group IDs (GIDs), and the implementation of the POSIX process model on the TPF system.
- See the *TPF C/C++ Language Support User’s Guide* for more information about the `chmod` function.
- See *TPF Concepts and Structures* for more information about special files.

ZFILE chown–Change the Owner and Group of a File or Directory

Use this command to change the user ID (UID), group ID (GID), or both of a file or directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE chown──┬──────┬── ──newowner──┬─────────────┬── ──path──────────────►◄
                 └──-R──┘               └─:newgroup───┘
```

-R recursively changes the owner or group for the specified file or directory. If you specify a directory, the owner or group of all files and subdirectories under that directory is changed. If you do not specify this parameter, the owner or group for only the specified file or directory is changed.

newowner
is the user ID (UID) or name of the new owner of the file.

newgroup
is the group ID (GID) or name of the new group of the file.

path
is the path name of the file or directory whose owner or group you are changing.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP chown
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.

- The ZFILE chown command is based on a subset of the chown utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	**Description**
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )

number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the user ID and group ID for `test.file` are changed.

```
User:   ZFILE chown 256:257 test.file

System: FILE0003I 15.26.53 chown 256:2... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

In the following example, the user name and group name for `test.directory` are changed.

```
User:   ZFILE chown -R root:bin test.directory

System: FILE0003I 15.26.59 chown -R ro... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE ls–List File and Directory Names and Attributes” on page 491 for information about how to display the owner and group of a file or directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and the *TPF C/C++ Language Support User’s Guide* for more information about user IDs, group IDs, and the implementation of the POSIX process model on the TPF system.

ZFILE cp–Copy a File or Directory

Use this command to copy a file or directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE cp──┬─┬──────┬──┬──────┬─ ──sourcefile── ──target────────┬──────►◄
              │ └──-f──┘  └──-p──┘                                  │
              └─┬──-r──┬──┬──────┬──┬──────┬─ ──source── ──target──┘
                └──-R──┘  └──-f──┘  └──-p──┘
```

-f
forces existing files to be overwritten regardless of the access permissions that are set.

Note: This parameter does not override the access permissions for a parent directory; that is, if the access permissions for the parent directory of the specified file do not allow write authority, the -f parameter is ignored and the specified file will not be copied. In addition, an error message is ***not*** issued if the copy is unsuccessful.

-p
preserves modification and access times, access permissions, and owning user and group IDs.

-r
recursively copies the specified file or directory. If you specify a directory, all files and subdirectories under that directory are copied. (This parameter is equivalent to the -R parameter.)

-R
recursively copies the specified file or directory. If you specify a directory, all files and subdirectories under that directory are copied. (This parameter is equivalent to the -r parameter.)

sourcefile
is the path name of the source file whose contents you are copying.

source
is the path name of the source file or directory whose contents you are copying.

target
is the path name of the file or directory to which you are copying the contents of the source file or directory. If you specify a directory, the contents of each source file or directory are copied to that directory with names specified by the last component of the source path name. If you specify a directory, the directory must exist. If you specify a file that does not exist, the file is created.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP cp
  ZFILE HELP

ZFILE ?

In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE cp -f -p** *source target*
  - **ZFILE cp -fp** *source target*
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE cp command is based on a subset of the cp utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  : preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (’ ’)
  : preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  : preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (′)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, `test.file` is copied to another directory called `/newdir`, creating a file with path name `/newdir/test.file`.

```
User:   ZFILE cp test.file /newdir

System: FILE0003I 15:31:22 cp test.fil... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE dd–Convert and Copy a File

Use this command to convert the contents of a file to a specified format and display the contents or copy the contents to another file.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE dd──┬──────────────┬──┬───────────────┬──┬───────────────────────┬───►◄
              └──if=inpath───┘  └──of=outpath───┘  └──conv=──┬─ascii───┬─┘
                                                             ├─ebcdic──┤
                                                             ├─lcase───┤
                                                             └─ucase───┘
```

if=*inpath*
: specifies the input file, where *inpath* is the path name of the file you are converting. If you do not specify this parameter, the input file is read from the standard input (`stdin`) stream.

of=*outpath*
: specifies the output file, where *outpath* is the path name of the file in which you want to store the data. If you do not specify this parameter, the data is written to the standard output (`stdout`) stream.

conv
: specifies the type of conversion, where:

  ascii
  : converts the data from EBCDIC to ASCII.

  ebcdic
  : converts the data from ASCII to EBCDIC.

  lcase
  : converts the data to lowercase.

  ucase
  : converts the data to uppercase.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP dd
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- On completion, the ZFILE dd command writes the number of input and output blocks to the standard error (`stderr`) stream, in the following format:

```
w.x records in
y.z records out
n bytes transferred in s secs (m bytes/sec)
```

Where:

w is the number of whole input blocks.

x is the number of partial input blocks. A partial input block is one that contains fewer bytes than specified by the input block size.

y is the number of whole output blocks.

z is the number of partial output blocks. A partial output block is one that was written with fewer bytes than specified by the output block size.

n is the number of bytes processed.

s is the number of seconds.

m is the number of bytes processed per second.

See "Examples" on page 456 for an example of this diagnostic message.

- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE dd command is based on a subset of the dd utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (″ ″)
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‛), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the contents of file `/etc/hello.txt` are converted to uppercase and displayed on a CRAS terminal. In addition, the number of bytes processed are written to the standard error (`stderr`) stream.

```
User:   ZFILE dd if=/etc/hello.txt conv=ucase

System: FILE0001I 11:41:45 START OF DISPLAY FROM dd if=/etc/hello.txt conv=ucase
        HELLO
        END OF DISPLAY
        FILE0002I 11.41.45 START OF ERROR DISPLAY FROM dd if=/etc/hello.txt conv=ucase
        0.1 records in
        0.1 records out
        6 bytes transferred in 1 secs (6 bytes/sec)
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE echo–Write Data to the Standard Output Stream

Use this command to write data to the standard output (`stdout`) stream. You can also use this command as a simple editor by using input/output (I/O) redirection.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE echo── ──string──────────────────────────────────────────►◄
```

string
: is the string to be written. End the string with a new-line character (the Enter key) or a redirection character (> or >>).

Separate parameters, such as *string* and redirection characters, by blanks. Follow the last parameter with a new-line character (press the Enter key).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP echo
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.

- The ZFILE echo command is based on a subset of the echo utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (\`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)

backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, string #!QZZ1 is written to `test.file`.

```
User:   ZFILE echo '#!QZZ1' > test.file

System: FILE0003I 15:31:22 echo '#!QZZ... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE export–Create or Display ZFILE Environment Variables

Use this command to create a new, permanent ZFILE environment variable, to display the environment variables that are currently defined, or to change the value associated with an existing environment variable.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
               ┌────────────────┐
               ▼                │
►►──ZFILE export──┬──────────────┬┴──────────────────────────►◄
                  └──name=value──┘
```

name
: is the name of the new ZFILE environment variable you want to create or the name of the variable that has a value you want to change.

value
: is the value to associate with the variable.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP export
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- If you do not specify any parameters, all ZFILE environment variables currently defined are displayed.
- Environment variables created with the ZFILE export command exist until removed with the ZFILE unset command or until you IPL the TPF system.
- The following environment variables are reserved for the TPF system:

  PWD Reserved for your current working directory. PWD is used and overwritten by the ZFILE cd command. Modifications to this variable will be lost without notification.

  OLDPWD
  Reserved for your previous working directory. OLDPWD is used and overwritten by the ZFILE cd command. Modifications to this variable will be lost without notification.

  HOME Reserved for your default directory. You can modify HOME to hold the desired default directory. The ZFILE cd command will use this default when a path is not specified. If HOME is removed using the ZFILE unset command, the ZFILE cd command will fail if you do not specify a path.

PATH Reserved as the default location to search for executable files and scripts. When using the ZFILE command, if the executable is not found in the current working directory, the TPF system searches the directory defined in the PATH environment variable. You can specify multiple directories by separating each with a colon (:).

- Environment variable names beginning with IBM and TPF are reserved for IBM use.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE export command is based on a subset of the export utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.

Examples

In the following example, the PATH environment variable is changed or created to contain multiple directories.

```
User:   ZFILE export PATH="/usr/bin:/usr/local"

System: FILE0003I 08:14:31 export PATH... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY.
```

In the following example, the environment variables that are currently available are displayed.

```
User:   ZFILE export

System: FILE0001I 08:14:31 START OF DISPLAY FROM export
        PWD="/usr/local"
        PATH="/usr/bin:/usr/local"
        HOME="/u/usrname"
        END OF DISPLAY
```

Related Information

- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See “ZFILE–Activate a TPF Segment or Script” on page 434 for more information about the ZFILE command.

- See “ZFILE cd–Change the Current Working Directory” on page 441 for more information about changing the current working directory.
- See “ZFILE unset–Unset Values of ZFILE Environment Variables” on page 546 for more information about removing environment variables.

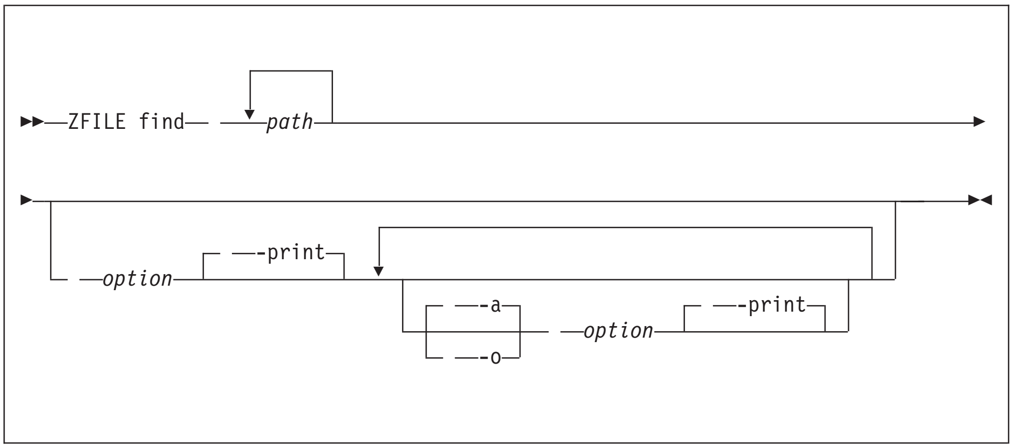
ZFILE find–Find a File

Use this command to find files in a specified directory or directory hierarchy.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



path
: is the path of the directory you want to search. If this directory contains other directories, those will be searched as well.

option
: specifies search criteria and consists of one or more of the following:

-exec *info* **{ } ;**
: allows you to take any found files and pass the information to another ZFILE command, where *info* is the command you want to pass information to, the braces ({ }) represent the found files, and the semicolon (;) indicates the end of the -exec option. The ending semicolon (;) must be delimited with a white space and escaped with a backslash (\). For example:

```
zfile find . \( -name tmp.* -o -name "*.t" \) -exec rm {} \;
```

first finds all files named `tmp` and all files with a `.t` extension. Those files are then passed to the ZFILE rm command (`rm`) and are subsequently removed.

-group *groupname*
: finds files having the same group owner, where *groupname* is the group owner you are searching for.

-name *pattern*
: finds file names matching the specified criteria, where *pattern* is the pattern you are searching for in the file names.

-perm *pppp*
: finds files matching the specified file permissions, where *pppp* is the permission setting in octal notation. By default, the -perm option finds only those files matching *pppp* exactly. To find files with permission settings of at

least *pppp*, add a minus sign (−) in front of *pppp*. For example, `zfile find . -perm −444` finds files with a permission value of 444 or higher. For a list of permission values, see "ZFILE chmod–Change the Access Permissions of a File or Directory" on page 444.

-type *char*
finds files having the same file type, where *char* is one of the following:

c Character special file.

d Directory.

f Regular file.

l Symbolic link.

p Named pipe.

s Socket file type.

-user *fileowner*
finds files having the same file owner, where *fileowner* is the file owner you are searching for.

-a connects two options with a logical AND.

-o connects two options with a logical OR.

-print
displays the matching files on the screen.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP find
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- The *option*, -a, and -o parameters can be grouped together using parentheses.
- An exclamation symbol (!) can be used before the *option* parameter to negate the search criteria. For example:
  – `find . ! -name "*.c" -a -perm 777` finds all files that do not have the .c extension, but have a permission setting of 777.
  – `find . -name "*.c" -a ! -perm 777` finds all files with the .c extension, but do not have a permission setting of 777.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.

- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE find command is based on a subset of the find utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (’ ’)
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example finds files with extensions `.c` and `.h` in or below your current directory.

```
User:   ZFILE find ./ -name '*.[ch]'

System: FILE0001I 08:14:31 START OF DISPLAY FROM find ./ -name '*.[ch]'
        ./abc.h
        ./file1.c
        ./file2.c
        ./xyz.h
        END OF DISPLAY
```

The following example finds all directories in or below your current directory.

```
User:   ZFILE find . -type d

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -type d
        ./directory1
        ./directory1/test
        ./directory2
        END OF DISPLAY
```

The following example finds file names that contain the string `qzz` and are owned by user `bill` in directory `work`.

```
User:   ZFILE find work -name "qzz*" -user bill

System: FILE0001I 08:14:31 START OF DISPLAY FROM find work -name "qzz*" -user ...
        work/apache/src/qzz1.c
        work/temp/qzz8.c
        END OF DISPLAY
```

The following example displays the names of all files with the name `tmp` or with the `.xx` extension in or below the current directory.

```
User:   ZFILE find . -name "tmp.*" -o -name "*.xx"

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -name tmp -o -name "*.xx"
        ./file1.xx
        ./tmp.c
        ./tmp.t
        END OF DISPLAY
```

The following example displays file names with access permission bit settings `777` (read, write, and execute permission for user, group, and other) in or below the current directory.

```
User:   ZFILE find . -perm 777

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -perm 777
        ./file1.c
        ./file2.c
        ./newdir/test.c
        END OF DISPLAY
```

The following example removes all files named `tmp` or files ending in `.xx` in or below the current directory.

```
User:   ZFILE find . \( -name "tmp.*" -o -name "*.xx" \) -exec rm {} \;

System: FILE0001I 08:14:31 find . \( -... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

The following example displays file names that do not end in `.o` or `.t` and are owned by user `bill` or are part of group `pgrms` in or below the current directory.

```
User:   ZFILE find . ! \( -name "*.o" -o -name "*.t" \) -a \( -user bill -o -group pgrms \)

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . ! \( -name "*.o" -o -name "...
        ./abc.c
        ./newdir/new.c
        ./test2.c
        END OF DISPLAY
```

Related Information

- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and the *TPF C/C++ Language Support User’s Guide* for more information about user IDs, group IDs, and the implementation of the POSIX process model on the TPF system.
- See “ZFILE chmod–Change the Access Permissions of a File or Directory” on page 444 for more information about the ZFILE chmod command.
- See “ZFILE rm–Remove a Link to a File or Directory” on page 518 for more information about the ZFILE rm command.

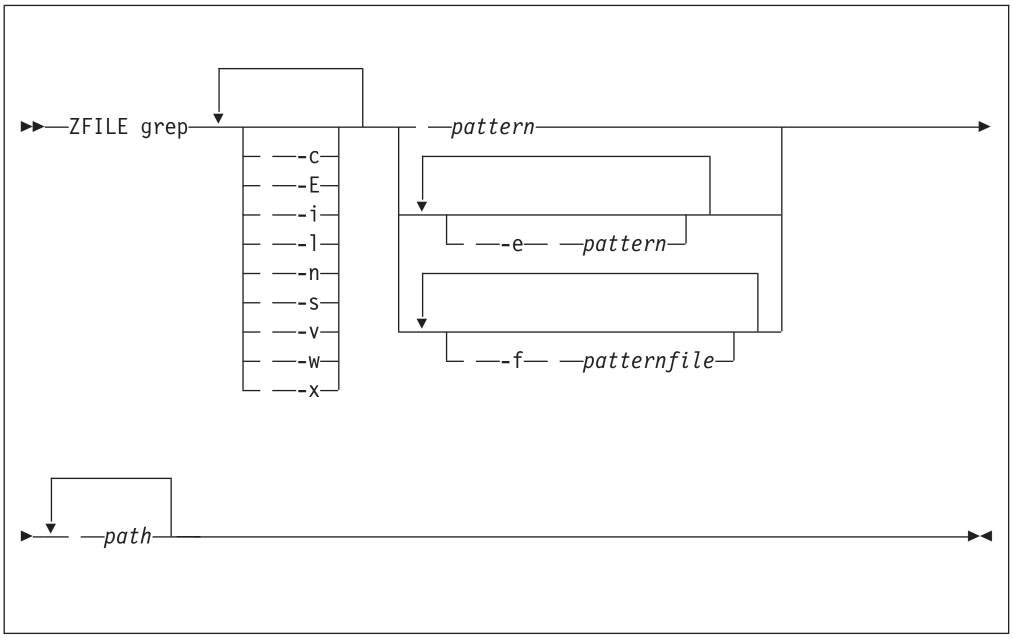
ZFILE grep–Search a File for a Specified Pattern

Use this command to search files for one or more patterns.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



-c displays the number of matching lines.

-E interprets *pattern* as an extended regular expression.

-i ignores the case when searching for matching lines.

-l lists the file names that contain matching lines.

-n displays the matching lines and the file line number.

-s suppresses the display of error messages that result from nonexisting or unreadable files.

-v displays the lines that do not match the specified criteria.

-w displays the lines that match whole words only.

-x displays lines only when *pattern* matches the entire line.

pattern
is the pattern, or regular expression, to use in searching the files.

-e allows you to specify more than one search pattern, each separated with a `-e`.

-f *patternfile*
reads one or more patterns from a file, where *patternfile* is the path to the file that contains the patterns you are searching for. Pattern strings within *patternfile* are separated by a new-line character (\n).

path
is the path to the source file whose contents are to be searched.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP grep
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE grep -c -i** *pattern path*
  - **ZFILE grep -ci** *pattern path*
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE grep command is based on a subset of the grep utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.

? Matches any single character.

[..] Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, all files with the `.h` extension are searched for regular expression `T.F`.

```
User:   ZFILE grep T.F *.h

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep T.F *.h
        bytecode.h: BGET_FREAD(ary, 256, 2);
        config.h:/* has_FSTATVFS:
        perl.h:#if defined(TPF) && defined(I_SYS_TIMES)
        END OF DISPLAY
```

In the following example, all files with the `.c` extension are searched for patterns `TPF` and `tpf_fork`.

```
User:   ZFILE grep -e TPF -e tpf_fork *.c

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep -e TPF -e tpf_fork *.c
        pp_sys.c:#ifdef TPF
        tpf.c:   struct tpf_fork_input fork_input;
        util.c:#if !defined(VMS) && !defined(TPF)
        END OF DISPLAY
```

In the following example, files `tpf.c` and `tpf.h` are searched for pattern `setenv`. Only file `tpf.c` contains the specified pattern.

```
User:   ZFILE grep -l setenv tpf.c tpf.h

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep -l setenv tpf.c tpf.h
        tpf.c
        END OF DISPLAY
```

In the following example, files beginning with either the letter `s` or the letter `t` and ending with either the `.c` or `.h` extension are searched for pattern `print`. The options indicate that the file name and the number of times that pattern is in that file will be displayed.

```
User:   ZFILE grep -c print* [st]*.[ch]

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep -c print* [st]*.[ch]
        sv.c:119
        sv.h:0
        taint.c:0
        tpf.c:2
        END OF DISPLAY
```

In the following example, all files are searched for pattern `has_fork`. The -i parameter indicates that the case in the pattern will be ignored.

```
User:   ZFILE grep -i has_fork *

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep -i has_fork *
        util.c:#ifdef HAS_FORK
        END OF DISPLAY
```

In the following example, file `util.c` is searched for patterns listed on separate lines in file `file1`. For this example, `file1` contains the following:

```
TPF
HAS_FORK
```

ZFILE grep

```
User:   ZFILE grep -f file1 util.c

System: FILE0001I 08:14:31 START OF DISPLAY FROM grep -f file1 util.c
        #if !defined(VMS) && !defined(TPF)  /* VMS' my_setenv() is in VMS.c */
        #if (!defined(DOSISH) || defined(HAS_FORK)
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE head–Write from the Beginning of a File

Use this command to write a specified number of records from the beginning of a file to the standard output (`stdout`) stream.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
                  ┌─ ──-n10──────┐
►►──ZFILE head────┼──────────────┼─ ──path──────────────────────────────────►◄
                  └─ ──-nnumline─┘
```

-n*numline*
: writes lines from the beginning of a file to the standard output (`stdout`) stream, where *numline* is the number of lines you want to write.

path
: is the path name of the file whose contents you are writing to the standard output (`stdout`) stream.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP head
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

You can redirect the standard output (stdout) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

You can redirect the standard error (stderr) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE head command is based on a subset of the head utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is /bin:/usr/bin:., the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering ZFILE echo '*' displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering ZFILE echo "\$PATH is \"$PATH\"" displays $PATH is "/bin:/usr/bin:."

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the first four records of `test.html` are displayed.

```
User:   ZFILE head -n4 test.html

System: FILE0001I 15:31:22 START OF DISPLAY FROM head -n4 test.html
        <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.0//EN">
        <HTML><HEAD>
        <TITLE>This is a test</TITLE>
        </HEAD><BODY>
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

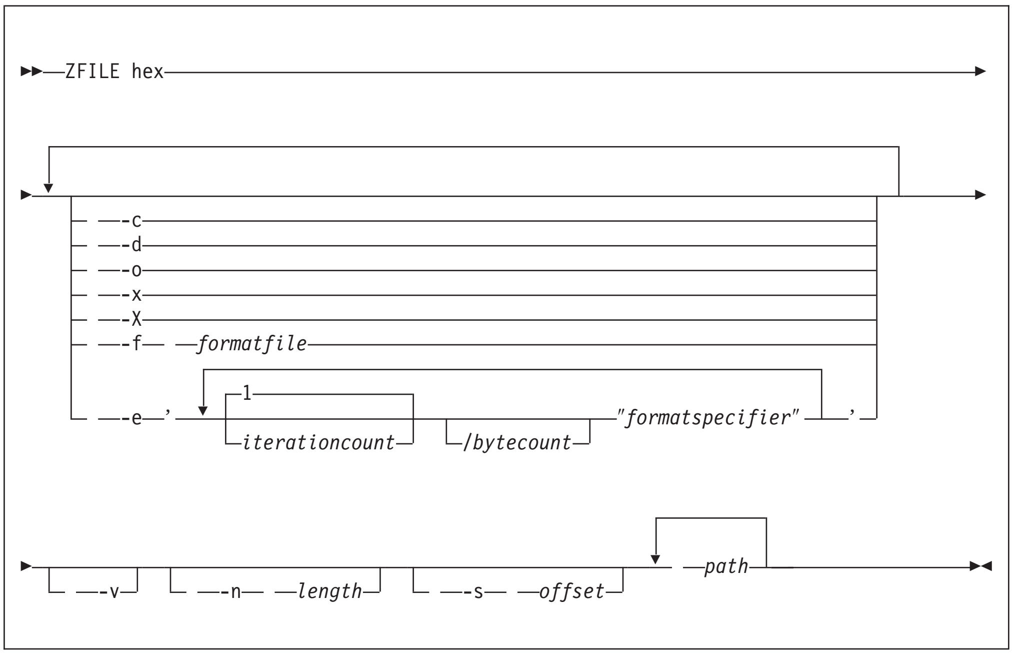
ZFILE hex–Display the Contents of a File

Use this command to write the contents of a file to the standard output (`stdout`) stream in character, decimal, hexadecimal, or octal format.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



-c displays a specified file in EBCDIC character display format. The displacement is shown on the left in hexadecimal, followed by 8 bytes of input data per line in EBCDIC with each character separated by spaces. Characters that cannot be printed are shown in hexadecimal format with the exception of the following C language escape sequences, which are displayed as *x* and have the following corresponding meanings:

\0 Null

\a Alert

\b Backspace

\f Form feed

\n New-line character

\r Carriage return

\t Horizontal tab

\v Vertical tab

-d displays a specified file in the decimal display format. The displacement is shown on the left in hexadecimal, followed by 8 bytes of input data per line in unsigned decimal format. Each byte is shown three columns wide and is padded on the left with zeros. The columns are separated by spaces.

-o displays a specified file in the octal display format. The displacement is shown on the left in hexadecimal, followed by 8 bytes of input data per line in octal. Each byte is shown three columns wide and is padded on the left with zeros. The columns are separated by spaces.

-x displays a specified file in a lowercase hexadecimal display format. The displacement is shown on the left in hexadecimal, followed by 8 bytes of input data per line. Each byte is shown two columns wide. The columns are separated by spaces.

-X displays a specified file in an uppercase hexadecimal display format. The displacement is shown on the left in hexadecimal, followed by 8 bytes of input data per line. Each byte is shown two columns wide. The columns are separated by spaces.

-v displays all lines of a file, including duplicate lines. By default, the ZFILE hex command suppresses identical lines of output showing repeated lines by displaying a single asterisk (*). The -v (verbose) parameter forces duplicate lines to be displayed.

-n *length*
specifies the number of bytes to display from a file, where *length* is the number of bytes to display. Specify the *length* as a decimal number or use one of the following letters immediately following the number to indicate a different specification:

k length is specified as a multiple of 1024

m length is specified as a multiple of 1 048 576

o length is specified in octal

x length is specified in hexadecimal

-s *offset*
specifies the number of bytes to skip in a file before the display begins, where *offset* is the number of bytes to skip. Specify the *offset* as a decimal number or use one of the following letters immediately following the number to indicate a different specification:

k offset is specified as a multiple of 1024

m offset is specified as a multiple of 1 048 576

o offset is specified in octal

x offset is specified in hexadecimal

path
is the path of the file you want to display. If multiple files are specified, the ZFILE hex command handles the input as one single file rather than individual files, and the concatenation of these files is in the order you specify.

-f *formatfile*
specifies that the *iterationcount*, *bytecount*, and *formatspecifier* are declared in a separate file, where *formatfile* is the path to the file that contains this format string information. In this file, blank lines and lines whose first nonblank character is a hash mark (#) are handled as comments and ignored. For more information about specifying the *iterationcount*, *bytecount*, and *formatspecifier*, see the -e parameter.

-e indicates that a format string is specified on the command line. The format string consists of one or more format units, and a format unit consists of an iteration count, a byte count, and a format specifier. The entire format string must be enclosed with single quotation marks.

iterationcount
is a positive integer that represents the number of times you want to have the *formatspecifier* parameter applied.

bytecount
is a positive integer that defines the number of bytes of input data interpreted each time the *formatspecifier* is applied. This number is always preceded by a slash (/). If you do not specify the *bytecount* parameter, the default is determined by the specified *formatspecifier*. (See Table 9 on page 480 for specific default values.) Do not specify the *bytecount* parameter if you are specifying a *formatspecifier* containing multiple conversion specifications (excluding `%_ad`, `%_ao`, `%_ax`, `%_aX`, `%_Ad`, `%_Ao`, `%_Ax`, and `%_AX`).

formatspecifier
is the information needed to determine how you want your data to be written to `stdout` and is enclosed by double quotation marks (" "). This parameter functions similarly to the format parameter of the `fprintf` function. (See *TPF C/C++ Language Support User's Guide* for more information about the `fprintf` function, including flags, widths, and precision specifications.) This parameter in the ZFILE hex command has the following limitations:

- An asterisk (*) cannot be used for width or precision.
- You must specify a *bytecount* or a precision for each `%s` conversion specification.
- The following conversion specifications are not supported: `%C`, `%h`, `%l`, `%n`, `%p`, `%q`, and `%S`.

Note: Conversion specifications always begin with a percent sign (%) as explained in the *TPF C/C++ Language Support User's Guide*.

The ZFILE hex command also supports C language escape characters (as described for the -c parameter) along with the following additional conversion specifications:

Table 7. Additional Conversion Specifications for the ZFILE hex Command

Type	Argument	Output Format
%_ad	Not applicable.	Input data displacement for each line shown in decimal notation. The displacement is cumulative across all specified input files.
%_ao	Not applicable.	Input data displacement for each line shown in octal notation. The displacement is cumulative across all specified input files.
%_ax	Not applicable.	Input data displacement for each line shown in lowercase hexadecimal notation. The displacement is cumulative across all specified input files.
%_aX	Not applicable.	Input data displacement for each line shown in uppercase hexadecimal notation. The displacement is cumulative across all specified input files.

Table 7. Additional Conversion Specifications for the ZFILE hex Command (continued)

Type	Argument	Output Format
%_Ad	Not applicable.	Ending displacement shown in decimal notation. The displacement is cumulative across all specified input files and is displayed once at the end of the display.
%_Ao	Not applicable.	Ending displacement shown in octal notation. The displacement is cumulative across all specified input files and is displayed once at the end of the display.
%_Ax	Not applicable.	Ending displacement shown in lowercase hexadecimal notation. The displacement is cumulative across all specified input files and is displayed once at the end of the display.
%_AX	Not applicable.	Ending displacement shown in uppercase hexadecimal notation. The displacement is cumulative across all specified input files and is displayed once at the end of the display.
%_c	Not applicable.	EBCDIC characters. Nonprintable characters are displayed in hexadecimal except for those that can be represented by standard C language escape sequences (see the -c parameter for more information).
%_p	Not applicable.	EBCDIC characters. Nonprintable characters are displayed as single periods (.).
%_e	Not applicable.	EBCDIC characters. Nonprintable characters are displayed in hexadecimal with the exception of some control characters, which are displayed with their corresponding 3-character name. Table 8 shows how the control characters will be displayed with this conversion specification.

When using the `%_e` conversion specification type, some control characters are displayed as 3-character names rather than in hexadecimal. The following table lists those control characters along with each corresponding 3-character name.

Table 8. Control Character Display for the _e Conversion Specification in the ZFILE hex Command

00 — `nul`	01 — `soh`	02 — `stx`	03 — `etx`
04 — `sel`	05 — `_ht`	06 — `rnl`	07 — `del`
08 — `_ge`	09 — `sps`	0a — `_lf`	0b — `_vt`
0c — `_ff`	0d — `_cr`	0e — `_so`	0f — `_si`
10 — `dle`	11 — `dcl`	12 — `dc2`	13 — `dc3`
14 — `res`	15 — `_nl`	16 — `_bs`	17 — `poc`
18 — `can`	19 — `_em`	1a — `ubs`	1b — `cul`
1c — `ifs`	1d — `igs`	1e — `irs`	1f — `itb`
20 — `_ds`	21 — `sos`	22 — `_fs`	23 — `wus`

Table 8. Control Character Display for the _e Conversion Specification in the ZFILE hex Command (continued)

24 — `byp`	25 — `lf_`	26 — `etb`	27 — `esc`
28 — `_sa`	29 — `sfe`	2a — `_sm`	2b — `csp`
2c — `mfa`	2d — `enq`	2e — `ack`	2f — `bel`
32 — `syn`	33 — `_ir`	34 — `_pp`	35 — `trn`
36 — `nbs`	37 — `eot`	38 — `sbs`	39 — `_it`
3a — `rff`	3b — `cu3`	3c — `dc4`	3d — `nak`
3f — `sub`	41 — `rsp`	ca — `shy`	e1 — `nsp`
ff — `_eo`			

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP hex
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- The default display for the ZFILE hex command displays a specified file in a format similar to the output of the ZDFIL command. The displacement is shown on the left in hexadecimal with a trailing dash followed by 16 bytes of input data per line, also in hexadecimal. These 16 bytes of input data are then repeated at the end of the line in character format with unprintable bytes shown as periods. The hexadecimal portion of the display is split into four 4-byte groups shown eight columns wide. The character representation is split into two 8-byte groups shown eight columns wide. The columns are separated by spaces.
- By default, the ZFILE hex command suppresses identical lines of output showing repeated lines by displaying a single asterisk (*). The -v (verbose) parameter forces duplicate lines to be displayed.
- If multiple input files are specified, the ZFILE hex command handles the input as one single file rather than individual files. The -n *length* and -s *offset* parameters apply to that concatenation and not the individual files.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  – **ZFILE hex -c -x** *path*
  – **ZFILE hex -cx** *path*
- The following table summarizes all the conversion specifications the ZFILE hex command supports with the default byte count shown in ***bold italics***, where ***n*** is a user-specified length:

Table 9. Summary of Supported Conversion Specifications for the ZFILE hex Command

Supported Conversion Specifications		Allowable Byte Counts				
Specifier	**Description**	**1**	**2**	**4**	**8**	***n***
_ad	Displacement in decimal					***n***
_ao	Displacement in octal					***n***
_ax	Displacement in lowercase hexadecimal					***n***
_aX	Displacement in uppercase hexadecimal					***n***

Table 9. Summary of Supported Conversion Specifications for the ZFILE hex Command (continued)

Supported Conversion Specifications		**Allowable Byte Counts**				
Specifier	**Description**	**1**	**2**	**4**	**8**	***n***
_Ad	End-of-file (EOF) displacement shown in decimal					***n***
_Ao	End-of-file (EOF) displacement shown in octal					***n***
_Ax	End-of-file (EOF) displacement shown in lowercase hexadecimal					***n***
_AX	End-of-file (EOF) displacement shown in uppercase hexadecimal					***n***
_c	EBCDIC display with nonprintable characters shown in C language standard escape notation or in hexadecimal	***1***				
_e	EBCDIC display with control characters shown using 3-character names or in hexadecimal	***1***				
_p	EBCDIC display with nonprintable characters shown as single periods (.)	***1***				
c	Unsigned character display	***1***				
d	Signed integer, decimal format display	1	2	***4***		
e	Double display, using [-]d.ddde±dd format			4	***8***	
E	Double display, using [-]d.dddE±dd format			4	***8***	
f	Double display, using [-]ddd.ddd format			4	***8***	
g	Similar to `e`; see the `fprintf` function in *TPF C/C++ Language Support User's Guide* for more details			4	***8***	
G	Similar to `E`; see the `fprintf` function in *TPF C/C++ Language Support User's Guide* for more details			4	***8***	
i	Signed integer display, decimal format	1	2	***4***		
o	Unsigned integer display, octal format	1	2	***4***		
s	String display (length must be specified)					***n***
u	Unsigned integer display, decimal format	1	2	***4***		
x	Unsigned integer display, lowercase hexadecimal format	1	2	***4***		
X	Unsigned integer display, uppercase hexadecimal format	1	2	***4***		

- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.

- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (\`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (\`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the default display format is shown, where file `filehex256` contains hexadecimal bytes 00 to ff. The display skips the first 192 bytes of this file.

```
User:   ZFILE hex -s 192 filehex256

System: FILE0001I 09:41:23 START OF DISPLAY FROM hex -s 192 filehex256
        000000c0- c0c1c2c3 c4c5c6c7 c8c9cacb cccdcecf {ABCDEFG HI......
        000000d0- d0d1d2d3 d4d5d6d7 d8d9dadb dcdddedf }JKLMNOP QR......
        000000e0- e0e1e2e3 e4e5e6e7 e8e9eaeb ecedeeef \.STUVWX YZ......
        000000f0- f0f1f2f3 f4f5f6f7 f8f9fafb fcfdfeff 01234567 89......
        00000100
        END OF DISPLAY
```

In the following example, the -f parameter is used to re-create the default ZFILE hex display format as shown in the previous example. File `filehex256` contains hexadecimal bytes 00 to ff. The display skips the first 192 bytes of this file. File `SampleFormatFile` contains the following three lines:

```
"%08.8_Ax\n"
"%08.8_ax" "- " 4/1 "%02.2x" " " 4/1 "%02.2x" " " 4/1 "%02.2x" " " 4/1 "%02.2x" " "
8/1 "%_p" " " 8/1 "%_p" "\n"
```

```
User:   ZFILE hex -s 192 -f SampleFormatFile filehex256

System: FILE0001I 09:41:23 START OF DISPLAY FROM hex -s 192 -f SampleFormatFile file...
        000000c0- c0c1c2c3 c4c5c6c7 c8c9cacb cccdcecf {ABCDEFG HI......
        000000d0- d0d1d2d3 d4d5d6d7 d8d9dadb dcdddedf }JKLMNOP QR......
        000000e0- e0e1e2e3 e4e5e6e7 e8e9eaeb ecedeeef \.STUVWX YZ......
        000000f0- f0f1f2f3 f4f5f6f7 f8f9fafb fcfdfeff 01234567 89......
        00000100
        END OF DISPLAY
```

ZFILE hex

In the following example, file `filehex256` contains hexadecimal bytes 00 to ff and is displayed using the lowercase hexadecimal format. The display skips the first 192 bytes of this file and shows only 16 bytes of data.

```
User:   ZFILE hex -x -n16 -s192 filehex256

System: FILE0001I 09:41:23 START OF DISPLAY FROM hex -x -n16 -s192 filehex256
        000000c0  c0  c1  c2  c3  c4  c5  c6  c7
        000000c8  c8  c9  ca  cb  cc  cd  ce  cf
        000000d0
        END OF DISPLAY
```

In the following example, file `filehex256` contains hexadecimal bytes 00 to ff and is displayed using both the EBCDIC character format and the lowercase hexadecimal character format. The display skips the first 192 bytes of this file and shows only 16 bytes of data.

```
User:   ZFILE hex -cx -n16 -s192 filehex256

System: FILE0001I 09:41:23 START OF DISPLAY FROM hex -cx -n16 -s192 filehex256
        000000c0   {   A   B   C   D   E   F   G
        000000c0  c0  c1  c2  c3  c4  c5  c6  c7
        000000c8   H   I  ca  cb  cc  cd  ce  cf
        000000c8  c8  c9  ca  cb  cc  cd  ce  cf
        000000d0
        END OF DISPLAY
```

Related Information

See “ZDFIL–Display File” on page 322 for more information about the ZDFIL command.

ZFILE kill–End a Process or Send a Signal

Use this command to end a process or send a signal.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- Because of the unique processing environment in the TPF system, only programs that are capable of receiving signals will be affected by this command.

Format

```
►►─ZFILE kill─┬─ ─-l──────────────────────┬──────────────────────►◄
              ├─ ─-s─ ─signal─ ─pidnum─┤
              └─ ─pidnum──────────────────┘
```

-l writes the names of all supported signals and their corresponding signal number to the standard output (`stdout`) stream.

-s sends a signal to the specified process identifier (ID).

signal
is the signal that you want to send. You can specify this value as one of the following integer values or the corresponding symbolic signal name:

Integer Value	Symbolic Signal Name
1	SIGHUP
2	SIGINT
3	SIGABRT
4	SIGILL
8	SIGFPE
9	SIGKILL
11	SIGSEGV
13	SIGPIPE
14	SIGALRM
15	SIGTERM
16	SIGUSR1
17	SIGUSR2
18	SIGABND
20	SIGCHLD
27	SIGIOER

pidnum
is the process ID created by the TPF system at the start of a process. If you do not specify the -s parameter, the SIGTERM signal is sent to end this process.

Use the `getpid` function in your application program to determine the process ID. The application program can be used to write the process ID to a file or to the console.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP kill
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE kill command is based on a subset of the kill utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (′)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, a list of supported symbolic signal names is displayed.

```
User:   ZFILE kill -l

System: FILE0001I 23.28.38 START OF DISPLAY FROM kill -l
         1) SIGHUP    2) SIGINT    3) SIGABRT   4) SIGILL
         8) SIGFPE    9) SIGKILL  11) SIGSEGV  13) SIGPIPE
        14) SIGALRM  15) SIGTERM  16) SIGUSR1  17) SIGUSR2
        18) SIGABND  20) SIGCHLD  27) SIGIOER
        END OF DISPLAY+
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE ln–Create a Link to a File

Use this command to create a link to a file.

Note: A *link* is either a hard link or a symbolic link. A *hard link* consists of one physical file and two or more i-nodes pointing to that file. The file is not physically deleted until the last link is deleted.

A *symbolic link* consists of one physical file, one i-node pointing to that file, and one or more i-nodes that specify the path to the physical file. The link can be to a regular file or to a directory. When a reference is made through a symbolic link, the TPF system follows the symbolic link chain until the i-node that points to the physical file is found. When the physical file is deleted, the links remain but return *file not found* errors if they are followed. If the physical file is re-created, the links point to the new file with no intervention.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- You cannot create multiple hard links to a directory. However, you can create multiple symbolic links.

Format

```
►►──ZFILE ln──┬──────┬──┬──────┬── ──source── ──link──────────────────►◄
              └─ -f ─┘  └─ -s ─┘
```

-f forces existing links to be removed so a new link can be created.

-s creates a symbolic link that points to the name of the file, not the file location. If you do not specify this parameter, a hard link is created.

source
is the path name of the file to which you want to create a link. This is the physical file, directory, or link that already exists.

link
is the path name of the file that will contain the link. This is the link that you want to create.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP ln
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE ln -f -s** *source link*
  - **ZFILE ln -fs** *source link*
- You can enter the ZFILE ls command with the -l parameter specified to display the hard link count and the symbolic links for a file or directory.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE ln command is based on a subset of the ln utility of the Portable Operating System Interface for Computer Environments (POSIX) standards. The -s parameter is an extension of the POSIX standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (′)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, a symbolic link is created in `newfile` that points to `olddir/file`.

```
User:   ZFILE ln -s olddir/file newfile

System: FILE0003I 15:31:22 ln -s olddi... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE ls–List File and Directory Names and Attributes

Use this command to display:

- A list of files or directories
- Information about a file
- Information about a directory as well as all the files and subdirectories under that directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
            ┌──────────┐   ┌─────────┐
            ▼ ┌ ──-C─┐ │   ▼         │
►►──ZFILE ls──┼──────┼─┴───┬─────────┴──────────────────►◄
              ├ ──-a─┤     └ ──path─┘
              ├ ──-d─┤
              ├ ──-F─┤
              ├ ──-i─┤
              ├ ──-l─┤
              ├ ──-q─┤
              ├ ──-R─┤
              ├ ──-r─┤
              ├ ──-t─┤
              └ ──-1─┘
```

-a lists all entries including those that start with a period (.), which are referred to as *hidden files*.

-C displays the list in a multiple column format.

-d displays information about the specified directory itself instead of information about the files and subdirectories under that directory.

-F indicates the file characteristics by putting one of the following characters after the file name, if applicable:

- **/** specifies a directory.
- **|** specifies a first-in-first-out (FIFO) special file, which is also referred to as a named pipe.
- @ specifies a symbolic link.
- ***** specifies an executable file.

-i displays the file serial (i-node) numbers.

-l displays the attributes of a file or directory in a long format, including the access permissions, number of links, user ID (UID), group ID (GID), file size, and time of the last modification. If you specify a directory, information about every file in that directory (one file per line) is displayed.

-q displays all nonprintable characters as a question mark (?).

-R recursively displays the contents of the specified directory and all subdirectories.

-r sorts the list in reverse of the usual order; you can combine this with other options that specify the sort order of the list.

-t sorts the list by the last modified time stamp.

-1 displays the list in a single-column format.

path
is the path name of the file or directory whose contents you are displaying. If you do not specify this parameter, the contents of the current working directory are displayed.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP ls
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- If you do not specify any parameters, only the file names are displayed.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE ls -l -a** *path*
  - **ZFILE ls -la** *path*
- If you specify parameters that are mutually exclusive (for example, -C and -l), the parameter that you specify last on the command is used.

  The following parameters are mutually exclusive:
  - -R and -d
  - -C and -l
  - -C and -1
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE ls command is based on the ls utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example lists the contents of the `/tmp` directory.

```
User:   ZFILE ls /tmp

System: FILE0001I 13:11:31 START OF DISPLAY FROM ls /tmp
        dir1               dummy.executable    dummy.readonly
        dir2               dummy.file
        END OF DISPLAY
```

The following example lists all the entries in the `/tmp` directory, including the hidden files.

```
User:   ZFILE ls -a /tmp

System: FILE0001I 13:11:31 START OF DISPLAY FROM ls -a /tmp
        .                  dir1               dummy.executable    dummy.readonly
        ..                 dir2               dummy.file
        END OF DISPLAY
```

The following example lists the attributes of all the files and subdirectories in the `/tmp` directory using the long format (-l). The first line of the display shows the total number of file system blocks occupied by files in the directory. The remainder of the display contains the following information:

- The file type and access permissions:
  - The first character identifies the file type as follows:

    - Regular file.

    c Character special file.

    d Directory.

    l Symbolic link.

    p FIFO special file (or named pipe).

    s Socket file type.
  - The next 9 characters indicate the access permissions in three groups of 3. The first group of 3 describes owner permissions; the second describes group permissions; the third describes other permissions. The following characters can be displayed:

    r Permission to read the file.

w Permission to write to the file.

S For owner permissions, the file is not executable (cannot be run) and the set-user-ID mode is set. For group permissions, the file is not executable and the set-group-ID mode is set.

s For owner permissions, the file is executable (can be run) and the set-user-ID mode is set. For group permissions, the file is executable and the set-group-ID mode is set.

x Permission to execute (or run) the file or search the directory.

- Permission is denied.

- The number of links to the file
- The name of the owner of the file or directory
- The name of the group that owns the file or directory
- For regular files, the size of the file expressed in bytes. For other kinds of files where the file size cannot be determined, the value is −1. For directories, the number of entries in the directory.
- The date and time that the file or directory was created (or for a file, the date and time the file was last changed)
- The name of the file or directory.

```
User:   ZFILE ls -l /tmp

System: FILE0001I 13:11:31 START OF DISPLAY FROM ls -l /tmp
        total 5
        drwxrwxrwx   1 root     system         4 May  5 09:30 dir1
        drwxrwxrwx   1 root     system         3 May  5 09:30 dir2
        -rwxr-x--x   1 apache   websrvr       12 May  5 09:34 dummy.executable
        -rwxrwxrwx   1 apache   websrvr       12 May  5 09:31 dummy.file
        -r--r--r--   1 apache   websrvr       12 May  5 09:35 dummy.readonly
        END OF DISPLAY
```

The following example lists the attributes for all the files and subdirectories in the root (/) directory, including the hidden files. See the previous example for an explanation of the display.

```
User:   ZFILE ls -la /

System: FILE0001I 10.19.37 START OF DISPLAY FROM ls -la /
        total 6
        drwxr-xr-x   0 root     bin            6 Jun  1 10:16 .
        drwxr-xr-x   0 root     bin            6 Jun  1 10:16 ..
        drwxr-xr-x   1 root     bin            6 May 29 06:04 dev
        drwxrwxrwx   1 nobody   tpfuser        3 Jun  1 10:16 etc
        drwxrwxrwx   1 nobody   tpfuser        5 Jun  1 10:18 tmp
        drwxrwxrwx   1 root     bin            7 May 29 14:41 usr
        END OF DISPLAY
```

The following example lists the file attributes for a file.

```
User:   ZFILE ls -l /tmp/dummy.file

System: FILE0001I 13:11:31 START OF DISPLAY FROM ls -l /tmp/dummy.file
        total 1
        -rwxrwxrwx   1 apache   websrvr       12 May  5 09:31 dummy.file
        END OF DISPLAY
```

The following example lists multiple files and directories.

ZFILE ls

```
User:   ZFILE ls /usr/aa /tmp/dir2 /tmp/dir1/a.html /tmp/dir1 /tmp/dir1/dir2

System: FILE0001I 13:11:31 START OF DISPLAY FROM ls /usr/aa /tmp/dir2 /tmp/dir1/a.ht...
        /tmp/dir1/a.html    /usr/aa

        /tmp/dir1/:
        a.html              dir2

        /tmp/dir1/dir2:
        a.exe

        /tmp/dir2:
        b.html              bb.html
        END OF DISPLAY
```

Related Information

- See “ZFILE chmod–Change the Access Permissions of a File or Directory” on page 444 for information about how to change the access permissions of a file or directory.
- See “ZFILE chown–Change the Owner and Group of a File or Directory” on page 448 for information about how to change the owner or group of a file or directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and the *TPF C/C++ Language Support User’s Guide* for more information about user IDs, group IDs, and the implementation of the POSIX process model on the TPF system.

ZFILE mkdir–Make a Directory

Use this command to make a directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- You cannot make a directory with the same name as an existing file in the new directory.

Format

```
                                              ┌─ ──┐
                                              ▼    │
►►──ZFILE mkdir──┬──────────────────────┬─ ───path─┴─────────────────────►◄
                 └─ ──-m── ──permission─┘
```

-m *permission*
: is the permission setting for the new directory specified in octal notation. Specify this permission setting by adding the following values:

4000
: has no effect on directories.

2000
: sets the effective group ID (GID) of the process that executes the directory. If you set this value for a directory, all files created in that directory will have the same GID of that directory.

0400
: allows the owner to read the directory.

0200
: allows the owner to write to the directory.

0100
: allows the owner to execute (or search) the directory.

0040
: allows the group to read the directory.

0020
: allows the group to write to the directory.

0010
: allows the group to execute (or search) the directory.

0004
: allows others to read the directory.

0002
: allows others to write to the directory.

0001
: allows others to execute (or search) the directory.

For example, to set the access permissions to allow the owner to read, write, and execute the directory, and to allow the group and others to read and execute the directory, specify 755 for *permission*.

Note: You do not have to specify the leading zeros for the access permission values.

path
is the path name of the new directory.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP mkdir
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can simplify the specification of the path name by using the ZFILE cd command to change the working directory of your TPF terminal.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE mkdir command is based on a subset of the mkdir utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double

quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‵), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (′)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, `my.new.directory` is created with search permission for the group.

```
User:   ZFILE mkdir 0010 my.new.directory

System: FILE0003I 15:31:22 mkdir 0010 ... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE cd–Change the Current Working Directory” on page 441 for more information about changing the current working directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

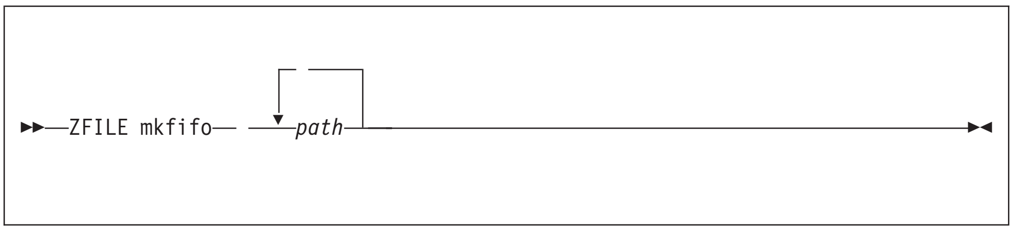
ZFILE mkfifo–Make a FIFO Special File

Use this command to make a FIFO special file. A *FIFO special file* is a file that is typically used to send data from one process to another so that the receiving process reads the data in first-in-first-out (FIFO) format. A FIFO special file is also known as a *named pipe*.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



path
  is the path name of the new FIFO special file.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP mkfifo
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- The ZFILE mkfifo command is based on a subset of the mkfifo utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can simplify the specification of the path name by using the ZFILE cd command to change the working directory of your TPF terminal.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, a FIFO special file named `my.new.fifo` is created. In the display from the ZFILE ls command, the size of the `my.new.fifo` file cannot be determined; therefore, a value of −1 is displayed in that field.

ZFILE mkfifo

```
User:   ZFILE mkfifo my.new.fifo

System: FILE0003I 14:20:15 mkfifo my.n... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY

User:   ZFILE ls -l -F

System: FILE0001I 14:20:35 START OF DISPLAY FROM ls -l -F
        total 3
        drwxrwxrwx   1 nobody   nogroup       10 Sep 30  1999 conf/
        drwxrwxrwx   1 root     bin            2 Sep 30  1999 htdocs/
        drwxrwxrwx   1 root     bin            3 Nov 16 11:54 logs/
        prw-r--r--   1 root     bin           -1 Mar 31 11:10 my.new.fifo|
        END OF DISPLAY
```

Related Information

- See “ZFILE ls–List File and Directory Names and Attributes” on page 491 for information about how to display information about files.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE mknod–Create a New Character Special File

Use this command to create a new character special file. Character special files allow you to access devices and other sources of data by defining device drivers and using the file system functions.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE mknod── ──path── ──c── ──major── ──minor──────────────────────────►◄
```

path
is the path name of the new character special file.

c indicates the special file is a character-oriented device.

major
is the major device number for the new character special file. The major device number specifies the device driver that the file system is to use for the character special file. Specify this parameter in the same way as a C language integer constant in octal, hexadecimal, or decimal format as follows:

- Any number that starts with 0 is octal; for example, 0765.
- Any number that starts with 0x is hexadecimal; for example, 0xFE.
- Any number that does not start with 0 or 0x is decimal; for example, 9876.

Major device numbers must be in the range 0–65535 (decimal), 0–0xFFFF (hexadecimal), or 0–0177777 (octal). Major device numbers from 32768 (0x8000, 0100000) to 65535 (0xFFFF, 0177777) are reserved for IBM use. Major device number 32768 (0x8000, 0100000) is reserved for regular files and cannot be used for character special files.

minor
is the minor device number for the new character special file. The minor device number provides information about the character special file to the device driver. The minor device number is meaningful only to the device driver that corresponds to the major device number. Specify this parameter in the same way as the *major* parameter. Minor device numbers must be in the range 0–65535 (decimal), 0–0xFFFF (hexadecimal), or 0–0177777 (octal).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP mknod
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- You must create a character special file to use a device driver.
- The major device number must correspond to a device driver index in segment UDDTBL (for user device drivers) or CDDTBL (for IBM device drivers). If no such device driver is installed, a 007515 system error will result if the character special file is opened.
- The access mode for the new character special file is set to 0666; that is, the owner, group, and others are authorized to read and write the file. You can change the access mode by entering the ZFILE chmod command or by calling the `chmod` function.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, a new character special file named `/dev/my.device` is created with major device number 2 and minor device number 0.

```
User:   zfile mknod /dev/my.device c 2 0

System: FILE0003I 15.19.34 mknod /dev/... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE ls–List File and Directory Names and Attributes” on page 491 for information about how to display information about files, including character special files.
- See the *TPF C/C++ Language Support User’s Guide* for more information about writing and installing device drivers, and about the `mknod` function.

ZFILE mv–Move or Rename a File or Directory

Use this command to move or rename a file or directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE mv──┬──────┬── ──source── ──target──────────────────────────►◄
              └─ ──-f┘
```

-f forces existing files to be overwritten regardless of the access permissions that are set.

Note: This parameter does not override the access permissions for a parent directory; that is, if the access permissions for the parent directory of the specified file do not allow write authority, the -f parameter is ignored and the specified file will not be moved or renamed. In addition, an error message is ***not*** issued if the move or rename is unsuccessful.

source
is the path name of the file or directory that you want to move or rename.

target
is the path name of the file or directory to which you are moving or renaming the file or directory. If you specify an existing directory, the contents of each source file are copied to that directory with names specified by the last component of the source path name. If you specify an existing file, the file is only overwritten if you specify the -f parameter.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP mv
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE mv command is based on a subset of the mv utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	**Description**
*	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, `old.name` is renamed.

```
User:   ZFILE mv old.name new.name

System: FILE0003I 15:31:22 mv old.name... COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

In the following example, `file.name` is moved to another directory.

```
User:   ZFILE mv file.name new.subdirectory/

System: FILE0003I 15:31:22 mv file.nam... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

In the following example, `old.name` is renamed and moved to another directory.

```
User:   ZFILE mv old.name new.subdirectory/new.name

System: FILE0003I 15:31:22 mv old.name... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE ps–Display Process Information

Use this command to display information about current processes, or entry control blocks (ECBs), in the system that were created by a `tpf_fork` function call.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
>>──ZFILE ps──┬───────────────────────────────────────────────────────┬──────────><
              │                ┌────────────────────────────────────┐ │
              │                ▼                                    │ │
              ├─┬──-f─┬────────┬──-G──realgrouplist──────────────┬─┴─┤
              │ └──-l─┘        │           ┌─,─────────────┐     │   │
              │                │           ▼               │     │   │
              │                ├──-o──────┬─field──────────┴─────┤   │
              │                │          ├─field=──────────┤    │   │
              │                │          └─field=heading───┘    │   │
              │                ├──-p──proclist───────────────────┤   │
              │                ├──-P──programlist────────────────┤   │
              │                ├──-t──termlist───────────────────┤   │
              │                ├──-u──effectiveuserlist──────────┤   │
              │                └──-U──realuserlist───────────────┘   │
              └──-L──────────────────────────────────────────────────┘
```

-f displays information about the ECBs using the full-listing format. The full-listing format includes the `PID`, `PPID`, `PROG`, `STIME`, `TIME`, and `USER` column headers. See Table 10 on page 511 for a description and summary of fields the ZFILE ps command supports.

-l displays information about the ECBs using the long-listing format. The long-listing format includes the `ADDR`, `PID`, `PPID`, `PROG`, `TIME`, `USER`, and `VSZ` column headers. See Table 10 on page 511 for a description and summary of fields the ZFILE ps command supports.

-G *realgrouplist*
displays information for specified ECBs, where *realgrouplist* is the list of real group identifiers (GIDs) for the ECBs you want information about. If more than one GID is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-o customizes the display based on what information you want to know about the ECBs.

field
is the name of the field to display, where *field* is any of the field names supported by the ZFILE ps command. See Table 10 on page 511 for a list of these names and a description of the information that field name will display. The table also includes information about default column headers and alias names of the display.

field=
indicates that there will be no column header for the field name specified in the *field* parameter. If all of the field names are specified with an equal sign (=), the column header line is omitted completely. See Table 10 on page 511 for default column header information.

field=*heading*
indicates that for the field name specified in *field*, the column header will be what is specified in *heading* rather than the default. See Table 10 on page 511 for default column header information.

-p *proclist*
displays information for specified ECBs, where *proclist* is the list of process identifier (process ID) numbers for the ECBs you want information about. If more than one process ID is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-P *programlist*
displays information for specified ECBs, where *programlist* is the list of 1- to 4-character program names for the ECBs you want information about. For example, `ZFILE ps -P cfj` will find all ECBs that have a program name that begins with `cfj`. This parameter is TPF-specific. If more than one program name is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-t *termlist*
displays information for specified ECBs, where *termlist* is the list of 3-byte hexadecimal terminal addresses for the ECBs you want information about. If more than one terminal address is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-u *effectiveuserlist*
displays information for specified ECBs, where *effectiveuserlist* is the list of effective user identifiers (UIDs) for the ECBs you want information about. If more than one UID is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-U *realuserlist*
displays information for specified ECBs, where *realuserlist* is the list of real user identifiers (UIDs) for the ECBs you want information about. If more than one UID is specified, they are separated with commas and the ZFILE ps command will connect them using a logical OR statement.

-L displays a list of valid field names for the -o *field* parameter. Each field name is shown on a separate line and is followed by valid alias names in parentheses (()). You can also refer to Table 10 on page 511 for the list of field names and their alias names.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP ps
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- Only processes created by a `tpf_fork` function and parent processes with active child processes are displayed.

- When entered without any parameters specified, the ZFILE ps command displays a listing of all current ECBs in the default format, which includes the `PID`, `PROG`, `ELAPSED` and `TIME` headers. See Table 10 for a summary of fields the ZFILE ps command supports.
- Only processes on the current central processing unit (CPU) are displayed.
- Process information is highly volatile and can change while the ZFILE ps command is running. The display may be out-of-date by the time it is shown.
- Items specified in -o *field*, -o *field*=, -o *field*=*heading*, -G *realgrouplist*, -p *proclist*, -P *programlist*, -t *termlist*, -u *effectiveuserlist*, and -U *realuserlist* can be separated by a comma or a blank. If blanks are used, the list of items must be enclosed in single quotation marks (' ') or double quotation marks (").
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  – **ZFILE ps -f -o** *field*
  – **ZFILE ps -fo** *field*
- The following table summarizes the supported display fields for the ZFILE ps command:

Table 10. Summary of Supported Display Fields for the ZFILE ps Command

Field Name	Default Header	Alias Names	Options	Description
addr	ADDR	sva	-l	The system virtual address (SVA) of the ECB.
etime	ELAPSED	elapsed	default	The elapsed time in [*dd*-]*hh*:*mm*:*ss* format.
group	GROUP	gname		The effective group identifier (GID). The decimal GID number is displayed if the group name is unavailable.
pgid	PGID			The parent group identifier in decimal format.
pid	PID		default, -f, -l	The process identifier in decimal format.
ppid	PPID		-f, -l	The parent process identifier in decimal format.
prog	PROG	pgm	default, -f, -l	The 4-character program name.
rgroup	RGROUP			The real group identifier. The decimal group identifier number (GID) is displayed if the group name is unavailable.
ruser	RUSER	runame		The real user identifier. The decimal user identifier number (UID) is displayed if the user name is unavailable.
stime	STIME	start, started	-f	The starting time of the process in [*Mmm dd*]*hh*:*mm*:*ss* format.

Table 10. Summary of Supported Display Fields for the ZFILE ps Command (continued)

Field Name	Default Header	Alias Names	Options	Description
time	TIME	cputime	default, -f, -l	The cumulative run time for the process in [*dd*-]*hh*:*mm*:*ss* format.
tty	TT	lniata, longtname, tname, tt		The 3-byte hexadecimal terminal address.
user	USER	uname	-f, -l	The effective user identifier. The decimal user identifier number (UID) is displayed if the user name is unavailable.
vsz	VSZ	memory, vsize	-l	The amount of virtual memory used for heap, stack storage, and TPF blocks combined, displayed as a decimal number in kilobytes (KB).

- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE ps command is based on a subset of the ps utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\\)
  : preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  : preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  : preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

    For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example shows the default display format.

```
User:   zfile ps

System: FILE0001I 19.03.30 START OF DISPLAY FROM ps
               PID PROG  ELAPSED     TIME
        1077215431 CFJP 00:00:00 00:00:00
        1077215446 CFIM 00:00:04 00:00:00
        1077215480 CFIX 00:00:01 00:00:00
        1076691213 CLTX 00:23:17 00:00:03
        1076625897 CLTW 00:23:58 00:00:02
        END OF DISPLAY
```

The following example shows the long-listing display format.

```
User:   zfile ps -l

System: FILE0001I 19.04.00 START OF DISPLAY FROM ps -l
        USER            PID       PPID PROG ADDR     VSZ     TIME
        nobody   1076691213 1076625897 CLTX 02397000 220 00:00:03
        root     1077215524 1077215531 CFJP 023DC000 208 00:00:00
        root     1077215531 1077215559 CFIX 023F1000 248 00:00:00
        root     1077215559          1 CFIM 02445000 248 00:00:00
        tpfdfltu 1076625897          1 CLTW 0262B000  68 00:00:02
        END OF DISPLAY
```

In the following example, multiple ECB selection criteria are used. ECBs with program name `cfix` or `cfim` are displayed. In addition, ECBs with an effective UID of `nobody` or `tpfdfltu` are displayed.

```
User:   zfile ps -l -P cfix,cfim -u nobody -u tpfdfltu
System: FILE0001I 16.26.39 START OF DISPLAY FROM ps -l -P cfix,cfim -u nobody -u tpf...
        USER            PID       PPID PROG ADDR     VSZ     TIME
        nobody   1078067228 1078002121 CLTX 014F4000 220 00:00:00
        root     1078133090          1 CFIM 018C6000 240 00:00:00
        tpfdfltu 1078002121          1 CLTW 019FB000  68 00:00:00
        root     1078133203 1078133090 CFIX 01A19000 244 00:00:00
        END OF DISPLAY
```

The following example shows a customized display. The requested output fields are `pgm` (an alias for `prog`), `vsz` (with a custom column heading of `MEMORY`), and `pid`. Only ECBs with a program name starting with `cfi` or a process ID of `1076625897` are displayed.

```
User:   zfile ps -o pgm,vsz=MEMORY,pid -P cfi -p 1076625897

System: FILE0001I 19.06.10 START OF DISPLAY FROM ps -o prog,vsz=MEMORY,pid -P cfi -p...
        PROG MEMORY        PID
        CFIX    256 1077280991
        CFIM    260 1077281006
        CLTW     68 1076625897
        END OF DISPLAY
```

The following example shows how the -L parameter works.

```
User:   zfile ps -L

System: FILE0001I 13.29.00 START OF DISPLAY FROM ps -L
        addr (sva)
        etime (elapsed)
        group (gname)
        pgid
        pid
        ppid
        prog (pgm)
        rgroup
        ruser (runame)
        stime (start, started)
        time (cputime)
        tty (lniata, longtname, tname, tt)
        user (uname)
        vsz (memory, vsize)
        END OF DISPLAY
```

Related Information

- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and *TPF C/C++ Language Support User's Guide* for more information about user identifiers (UIDs), group identifiers (GIDs), and the implementation of the POSIX process model on the TPF system.
- See *TPF C/C++ Language Support User's Guide* for more information about the `tpf_fork` function, process identifiers (process IDs), and parent process IDs.

ZFILE pwd–Write the Current Working Directory

Use this command to write the path name of the current working directory to the standard output (`stdout`) stream.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE pwd───────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP pwd
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE pwd command is based on a subset of the pwd utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (’ ’)
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (″ ″)
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the current working directory is displayed.

```
User:   ZFILE pwd

System: FILE0001I 15:31:22 START OF DISPLAY FROM pwd
        /tmp
        END OF DISPLAY
```

Related Information

- See “ZFILE cd–Change the Current Working Directory” on page 441 for information about changing the current working directory.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE rm–Remove a Link to a File or Directory

Use this command to remove a link to a file or a directory from the file system.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- A directory must be empty before it can be removed unless you specify the -r or -R parameter.

Format

```
              ┌────────────┐    ┌─  ──┐
              ▼            │    ▼     │
►►──ZFILE rm──┬────────────┴─  ──path──┴─ ─────────────────────────►◄
                ├─ ──-f─┤
                ├─ ──-r─┤
                └─ ──-R─┘
```

-f forces the removal of the specified file or directory regardless of the access permissions that are set.

Note: This parameter does not override the access permissions for a parent directory; that is, if the access permissions for the parent directory of the specified file do not allow write authority, the -f parameter is ignored and the specified file will not be removed.

-r recursively removes the specified file or directory. If you specify a directory, all files and subdirectories under that directory are removed. (This parameter is equivalent to the -R parameter.)

-R recursively removes the specified file or directory. If you specify a directory, all files and subdirectories under that directory are removed. (This parameter is equivalent to the -r parameter.)

path
is the path name of the file or directory that you are removing from the file system.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP rm
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE rm -f -r** *path*

– **ZFILE rm -fr** *path*

- You can remove directories as well as other files. The directory that you are removing must be empty unless you specify the -r or -R parameter.
- You can also use the ZFILE rmdir command to remove directories
- Removing a file is not necessarily the same as deleting a file; a link to a file is actually removed. A file can have multiple links and is not actually deleted until its final link is removed.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE rm command is based on a subset of the rm utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (’)
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example removes a file.

```
User:   ZFILE rm my.file

System: FILE0003I 13.12.45 rm my.file COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

The following example removes a directory and recursively removes any files or subdirectories contained in the directory.

```
User:   ZFILE rm -rf my.directory

System: FILE0003I 15.10.13 rm -rf my.d... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE ls–List File and Directory Names and Attributes” on page 491 for information about how to display a list of the files and directories in the file system.
- See “ZFILE rmdir–Remove a Directory” on page 521 for more information about removing directories.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See *TPF Application Programming* and the *TPF C/C++ Language Support User’s Guide* for more information about user IDs, group IDs, and the implementation of the POSIX process model on the TPF system.

ZFILE rmdir–Remove a Directory

Use this command to remove a directory.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.
- The directory must be empty before it can be removed. Use the ZFILE rm command to remove any files before entering the ZFILE rmdir command.

Format

```
                      ┌─    ──┐
                      ▼       │
>>──ZFILE rmdir──  ────path───┴──────────────────────────────────><
```

path
  is the path name of the directory you want to remove.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP rmdir
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE rmdir command is based on a subset of the rmdir utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.

? Matches any single character.

[..] Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the `misc` directory is removed.

```
User:   ZFILE rmdir misc

System: FILE0003I 15:31:22 rmdir misc COMPLETED SUCCESSFULLY.  NO OUTPUT TO DISPLAY
```

Related Information

- See “ZFILE rm–Remove a Link to a File or Directory” on page 518 for more information about removing files.
- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE sed–Noninteractive Stream Editor

Use this command to apply editing subcommands to a file and to display the results on the screen. Once the editing script is completed, the results are written to the standard output (`stdout`) stream, but the original file remains unchanged.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►──ZFILE sed──┬──────┬────────────────────────────────────────────────►
               └──-n──┘

►──┬──────────────────────────────────────────────────────────────┬────►
   │ ┌──────────────────────────────────────────────────────────┐ │
   │ ▼                                                          │ │
   ├───┬──-e──┬───────────┬──subcmd──┬─────────────────────┬─┬──┴─┤
   │   │      │ ┌─,─────┐ │          └─┬──────┬──argument──┘ │    │
   │   │      │ ▼       │ │            └──-e──┘              │    │
   │   │      └─address─┘                                    │    │
   │   └──-f──scriptfile───────────────────────────────────┘      │
   │   ┌─,─────┐                                                  │
   │   ▼       │                                                  │
   └───┬─────────┬──subcmd────────────────────────────────────────┘
       └─address─┘

   ┌──────────┐
   ▼          │
►──┬──────────┬──────────────────────────────────────────────────────►◄
   └──path────┘
```

Note: Throughout this section, the term *script* refers to a combination of *address*, *subcmd* and, where necessary, *argument*.

-n
displays only the lines that have been coded to print through the p or P editing subcommands. If you do not specify this parameter, all lines of the file you are editing will be written to the standard output (`stdout`) stream.

-e
allows you to specify more than one piece of input on the command line. You must use the -e parameter when using more than one editing subcommand, an editing subcommand with a *scriptfile*, and when the specified editing subcommand requires additional input that must be placed on a new line. When entering the editing script in *scriptfile* rather than on the command line, replace -e with a new line.

-f *scriptfile*
runs a set of editing subcommands that are stored in a separate file, where *scriptfile* is the path to the file that contains the editing script you want to run. The script in *scriptfile* contains a combination of subcommands and the

appropriate addresses and arguments. See the *subcmd* parameter description for more information about the subcommands you can use in *scriptfile*.

path
: is the path to the file you want to read and edit. The results of the editing script are written to `stdout`, but the original file remains unchanged. If multiple files are specified, the files will be read in the order specified and the result will be written as one concatenated file to `stdout`.

address
: is the address of the line in the specified file the editing subcommand applies to. If an editing subcommand allows up to two addresses, you can specify no address, one address, or two addresses. If no address is specified, the editing subcommand is applied to each line of the specified file. If one address is specified, the editing subcommand applies only to the line matching that address. If two addresses are specified, the addresses are separated by a comma and are interpreted as a range. Each address can be constructed using the following:

  n specifies the line number to which the editing subcommand applies. If the editing subcommand accepts more than one address, you can supply two line numbers to indicate a range. For example `4,6` indicates that you want to edit lines 4 to 6.

  $ specifies the last line of the specified file.

  /*regexp***/**
  : specifies an address based on the specified regular expression. All regular expressions must be delimited, which is usually done with a slash (/).

argument
: is additional information required by the editing subcommand. See the *subcmd* parameter description for more information about necessary arguments.

subcmd
: is the editing subcommand to apply to the addressed line in the specified file. The ZFILE sed command reads the specified file line by line into a special area known as the *pattern buffer*. Once a line is in the pattern buffer, its address is compared to the specified address and, if there is a match, the editing subcommand is performed. By default, each line is then written to `stdout` as the next line of the specified file is read into the pattern buffer, but the file specified in *path* is not changed. Some editing subcommands use a second area known as the *hold space*, which serves as temporary storage.

  The following table summarizes all the allowable editing subcommands. The columns are labeled as follows:

  subcmd
  : is the subcommand.

  Description
  : describes what the subcommand does.

  Argument Information
  : provides information about arguments that need to be supplied.

  Number of Allowable Addresses
  : indicates the number of addresses that are allowed for that particular subcommand.

  Script Syntax
  : shows the syntax of the *address*, *subcmd* and, optionally, *argument*

(collectively known as the script), where [0addr], [1addr], or [2addr] represent the maximum number of addresses allowed and *text* or *label* represents the required argument information. (The square brackets ([]) shown in the Script Syntax column are not coded with the addresses.)

In this table, the syntax of each subcommand is shown as it would appear in *scriptfile*. When using these subcommands on the command line, refer to the syntax diagram for the location of the necessary -e, which is replaced by a new line when using these subcommands in *scriptfile*.

Table 11. ZFILE sed Command Editing Subcommand

subcmd	**Description**	**Argument Information**	**Number of Allowable Addresses**	**Script Syntax**
a\	Appends lines of text to the specified file. The ZFILE sed command writes the text after completing all other specified editing subcommands for that line and before reading the next line.	*text* is the text you want to add. The first line that does not end with a backslash (\) character is the end of the text you are appending, but the \ is not handled as part of the text.	0 or 1	[1addr]a\ *text*
b	Branches to the specified *label*. If no label is specified, the branch goes to the end of the script. When specifying a label, do not put any additional white space after the label.	*label* is the location in which to branch. See the label subcommand for more information.	0, 1, or 2	[2addr]b *label*
c\	Replaces the pattern buffer with the supplied text.	*text* is the text you want to use in the replacement.	0, 1, or 2	[2addr]c\ *text*
d	Deletes the contents of the pattern buffer. The script is restarted on the next line of the specified file.	No arguments allowed.	0, 1, or 2	[2addr]d
D	Deletes the information held in the pattern buffer up to and including the first new-line character (\n). The script is then restarted from the beginning and is applied to the text in the pattern buffer.	No arguments allowed.	0, 1, or 2	[2addr]D
g	Replaces the pattern buffer with the contents of the hold space.	No arguments allowed.	0, 1, or 2	[2addr]g
G	Appends the pattern buffer with a new-line character (\n) and then the contents of the hold space without replacing the entire line.	No arguments allowed.	0, 1, or 2	[2addr]G
h	Copies the pattern buffer into the hold space. Any text previously held is replaced.	No arguments allowed.	0, 1, or 2	[2addr]h

Table 11. ZFILE sed Command Editing Subcommand (continued)

subcmd	**Description**	**Argument Information**	**Number of Allowable Addresses**	**Script Syntax**
H	Appends the hold space with a new-line character (\n) and then the text of the pattern buffer. No text is replaced.	No arguments allowed.	0, 1, or 2	[2addr]H
i\	Inserts the specified *text*.	*text* is the text to insert.	0 or 1	[1addr]i\ *text*
l	Writes the pattern buffer to `stdout` so nonprintable characters are visible. The end of a line is represented by the dollar sign character ($), and the characters \\, \a, \b, \f, \r, \t, and \v are printed as escape sequences. Other nonprintable characters will be written as a 2–digit hexadecimal number for each byte.	No arguments allowed.	0, 1, or 2	[2addr]l
n	Skips the pattern buffer and moves to the next input line. If the -n parameter is not specified, the skipped line will be printed to `stdout`. For example, this parameter could be used to delete every third line of the file before writing it to `stdout` by entering `ZFILE sed -e 'n' -e 'n' -e 'd' /u/file1` on the command line.	No arguments allowed.	0, 1, or 2	[2addr]n
N	Appends the next line of input to the current line in the pattern buffer with a new-line character (\n) separating the original material and the appended material. As a result, the current line number changes.	No arguments allowed.	0, 1, or 2	[2addr]N
p	Prints the text in the pattern buffer to `stdout`. When the -n parameter is not specified, this is the default.	No arguments allowed.	0, 1, or 2	[2addr]p
P	Similar to the p subcommand except that it prints the text in the pattern buffer only up to and including the first new-line character (\n).	No arguments allowed.	0, 1, or 2	[2addr]P

Table 11. ZFILE sed Command Editing Subcommand (continued)

subcmd	**Description**	**Argument Information**	**Number of Allowable Addresses**	**Script Syntax**
q	Quits the ZFILE sed command by skipping the remainder of the script and reading no more input lines	No arguments allowed.	0 or 1	[1addr]q
r	Reads text in the specified *argfile* and writes that text to stdout before reading the next input line. If *argfile* does not exist or cannot be read, it is handled as an empty file.	*argfile* is the path to the file that contains the text you want read at the specified address.	0 or 1	[1addr]r *argfile*
s	Substitutes *old* text with *new* text.	Indicates which pieces of text you want to remove (*old*) as well as what you want put in its place (*new*). By default, the substitution will only take place on the first occurrence of *old* on each line. You can change this with a combination of following options: *n* substitutes only the n^{th} occurrence of *old* per line. This option is not compatible with the g option. **g** replaces all instances of *old* throughout the file. This option is not compatible with the *n* option. **p** writes the changed file to `stdout` only if a successful substitution occurs. **w** *file* writes the results of the script to the end of *file* if a substitution occurs, where *file* is the path of the file in which to save the results of the script.	0, 1, or 2	[2addr]s/*old*/*new*/[gnp][w*file*] You can use any single printable character, other than a space or new-line character (\n), instead of a slash (/) to delimit *old* and *new*. The delimiter itself may appear as a literal character in *old* or *new* if it is preceded with a backslash (\).
t	Branches to the indicated *label* only if a successful substitution has occurred since either reading the last input line or running the last t subcommand. If you do not specify a label, the branch goes to the end of the script.	*label* is the location in which to branch. See the label subcommand for more information.	0, 1, or 2	[2addr]t *label*

Table 11. ZFILE sed Command Editing Subcommand (continued)

subcmd	Description	Argument Information	Number of Allowable Addresses	Script Syntax
w	Writes the pattern buffer to the end of *file*.	*file* is the path to the file to append.	0, 1, or 2	[2addr]w *file*
x	Exchanges the text in the hold space with the contents of the specified line.	No arguments allowed.	0, 1, or 2	[2addr]x
y	Replaces any character that occurs in *string1* with the corresponding character in *string2* on the specified line.	Specify *string1* and *string2*, where *string1* is the string to search for and *string2* is the replacement string. The strings must be the same length.	0, 1, or 2	[2addr]y/*string1*/*string2* You can use any character other than a backslash or new-line character (\n) instead of the slash to delimit the strings.
=	Includes the decimal value of the current line number when writing to `stdout`.	Not applicable	0 or 1	[1addr]=
Additional Tools for Creating an Editing Script				
{*subcmd*}	The braces ({}) allow you to group within your script, where *subcmd* is the group of editing subcommands to group together. The ZFILE sed command runs the script as a group only on those lines that match the specified address.	Not applicable.	0, 1, or 2	[2addr]{*subcmd*}
!	Runs the specified subcommands only on those lines not matching the specified address.	*subcmd* is the editing subcommand you want to run on the lines not matching the specified address.	Determined by which editing sub-command you are using.	[2addr]!*subcmd*
#	Allows you to code comments into your script. The single exception is when `#n` is on the first line in *scriptfile*, which is interpreted to mean the same as the -n parameter on the command line. An empty script line is also handled as a comment.	Not applicable.	Not applicable.	[0addr]# *comment*
label	Designates the name of the location in which to branch and must be preceded by a colon (:).	*label* is used with either the b subcommand or the t subcommand. See the b and t subcommands for more information.	Not applicable.	[0addr]: *label*

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

ZFILE HELP sed
ZFILE HELP
ZFILE ?

In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE sed -n -f** *scriptfile path*
  - **ZFILE sed -nf** *scriptfile path*
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE sed command is based on a subset of the sed utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  : preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  : preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (″ ″)
  : preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

  For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

  If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

  In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example appends the text `Sandy Davis, 954 East Harvard Drive, Denver, CO` after the third line of the file named *list*, which contains the following:

```
Emily Adams, 256 Middletown Road, Plymouth, MA
John Baker, 333 Union Road, Plymouth, MA
Alice and David Smith, 345 East Broadway, Richmond, VA
Brian Adams, 2000 Bayshore Court, San Diego, CA
Sally Jones, 7333 West 6th Street, Boston, MA
```

```
User:   ZFILE sed -e '3a\' -e 'Sandy Davis, 954 East Harvard Drive, Denver, CO' list

System: FILE0001I 08:14:31 START OF DISPLAY FROM sed -e '3a\' -e 'Sandy Davis, 954 E...
        Emily Adams, 256 Middletown Road, Plymouth, MA
        John Baker, 333 Union Road, Plymouth, MA
        Alice and David Smith, 345 East Broadway, Richmond, VA
        Sandy Davis, 954 East Harvard Drive, Denver, CO
        Brian Adams, 2000 Bayshore Court, San Diego, CA
        Sally Jones, 7333 West 6th Street, Boston, MA
        END OF DISPLAY
```

The following example deletes line 2 to line 4 from file `list`. The contents of `list` are shown in the previous example.

```
User:   ZFILE sed '2,4d' list

System: FILE0001I 08:14:31 START OF DISPLAY FROM sed '2,4d' list
        Emily Adams, 256 Middletown Road, Plymouth, MA
        Sally Jones, 7333 West 6th Street, Boston, MA
        END OF DISPLAY
```

The following example substitutes the word `Massachusetts` for the letters `MA` in `list`. The contents of file `list` are shown in the first example.

```
User:   ZFILE sed 's/MA/Massachusetts/' list

System: FILE0001I 08:14:31 START OF DISPLAY FROM sed 's/MA/Massachusetts/' list
        Emily Adams, 256 Middletown Road, Plymouth, Massachusetts
        John Baker, 333 Union Road, Plymouth, Massachusetts
        Alice and David Smith, 345 East Broadway, Richmond, VA
        Brian Adams, 2000 Bayshore Court, San Diego, CA
        Sally Jones, 7333 West 6th Street, Boston, Massachusetts
        END OF DISPLAY
```

The following example counts the number of lines in file `list`. The -n parameter suppresses the automatic output of each input line. The contents of file `list` are shown in the first example.

```
User:   ZFILE sed -n '$=' list

System: FILE0001I 08:14:31 START OF DISPLAY FROM sed -n '$=' list
        5
        END OF DISPLAY
```

The following example runs a script located in `scriptfile` against file `list2`. The script, which uses the N, b, d, and s subcommands, joins all lines ending with a backslash (\) and deletes comment and blank lines. The contents of `list2` are:

```
# text file to show joining lines
# also contains blank lines and comments
Emily Adams, 256 Middletown Road, Plymouth, MA

John Baker, 333 Union Road, Plymouth, \
MA
Alice and David Smith, 345 East Broadway, Richmond VA

Brian Adams, 2000 Bayshore Court, San Diego \
CA
Sally Jones, 7333 West 6th Street, Boston, MA
```

The contents of `scriptfile` are:

```
#   /^#/d   delete all lines beginning with '#' (comments?)
#   /^$/d   delete all empty lines (/./!d could be used instead)
#   :join
#   /\\$/{
#        N
#        s/\\\n//
#        bjoin
#   }
#   would join all lines ended with '\', after deleting
#   the '\' itself
#

    /^#/d
/^$/d
:join
/\\$/{
        N
        s/\\\n//
        bjoin
}
```

```
User:   ZFILE sed -f scriptfile list2

System: FILE0001I 08:14:31 START OF DISPLAY FROM sed -f scriptfile list2
        Emily Adams, 256 Middletown Road, Plymouth, MA
        John Baker, 333 Union Road, Plymouth, MA
        Alice and David Smith, 345 East Broadway, Richmond, VA
        Brian Adams, 2000 Bayshore Court, San Diego, CA
        Sally Jones, 7333 West 6th Street, Boston, MA
        END OF DISPLAY
```

The following example runs a script located in `scriptfile2` against file `index.html`. The purpose of the script is to look through a file and display all the URLs referenced in the file. The contents of `scriptfile2` are:

```
#
# list URLs and associated comments in HTML
#
# NOTES
# 1. Results are formatted as: URL<tab>comment
# 2. ALT values spanning more than one line have leading whitespace reduced
#    to a single space.
#


:top

# imagemap links...
/<[Aa][Rr][Ee][Aa]/ {
        :imagemap
        ## read up to end of directive
        />/ !{
                N
                s/[     ]*\n[   ]*/ /
                b imagemap
        }

        ## move remainder of line into hold space
        h
        x
        s/[^>]*>//
        x
        s/>.*/>/

        ## reformat directive as: <href=foo alt="blah">
        s/ [Aa][Ll][Tt]=/ alt=/
        s/ [Hh][Rr][Ee][Ff]=/ href=/
```

```
        # ensure ALT attribute is present
        / alt=/ ! s/>/ alt="">/
        # ensure both attributes are quoted
        s/ alt=\([^" >]\{1,\}\(/ alt="\1"/
        s/ href=\([^" >]\{1,\}\(/ href="\1"/
        # ensure HREF precedes ALT
        s/\( alt="[^"]*"\(\(.*\(\( href="[^"]*"\(>/\3\2\1>/
        s/^[^<]*//

        # print result as: URL<tab>comment
        s/.*href="\([^"]*\(" alt="\([^"]*\(">/\1        \2/p

        # continue processing with remainder of line
        s/.*//
        x
        b top
}

## anchor links...
/<[Aa][^a-zA-Z0-9]\{0,1\}/ {
        :anchor
        # read up to closing tag
        /<\/[Aa]>/ !{
                N
                s/[     ]*\n[   ]*/ /
                b anchor
        }

        s/"//g
        G
        s/<\/[Aa]>\(.*\(\(.\($/\2\1/
        # s/^[^<]*\(<[Aa] \(/\1/

        # only continue if this anchor contains HREF
        /<[Aa][^>]* [Hh][Rr][Ee][Ff]=.*\n/ ! {
                s/.*\n//1
                b top
        }

        s/.* [Hh][Rr][Ee][Ff]=\([^>]*\(>[       ]*\(.*\n\(/\1   \2/
        s/<[^>]*>//g
        P
        s/.*\n//1
        b top
}
```

```
User:   zfile sed -nf url.t /usr/local/apache/htdocs/index.html

System: FILE0001I 08.05.41 START OF DISPLAY FROM sed -nf url.t /usr/local/apache/htd...
        http://www.apache.org/.    Apache..
        manual/index.html.    documentation..
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE tail–Write from the End of a File

Use this command to write a specified number of records from the end of a file to the standard output (`stdout`) stream.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
                      ┌─ ──-n10────────┐      ┌─  ───┐
►►──ZFILE tail────────┼────────────────┼─ ────▼path──┴──────────────────►◄
                      ├─ ──-cnumchar───┤
                      └─ ──-nnumline───┘
```

-c*numchar*
: writes characters from the end of a file to the standard output (`stdout`) stream, where *numchar* is the number of characters you want to write.

-n*numline*
: writes lines from the end of a file to the standard output (`stdout`) stream, where *numline* is the number of lines you want to write.

path
: is the path name of the file whose contents you are writing to the standard output (`stdout`) stream.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP tail
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.

- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE tail command is based on a subset of the tail utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).
- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (' ')
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

  double quotation marks (" ")
  preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign

($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the last four records of `test.file` are displayed.

```
User:   ZFILE tail -n4 test.file

System: FILE0001I 15:31:22 START OF DISPLAY FROM tail -n4 test.file
        <TITLE>This is a test</TITLE>
        </HEAD><BODY>
        <B>Test</B>
        </BODY></HTML>
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE tee–Duplicate the Output Stream

Use this command to copy standard input (`stdin`) and write it to both standard output (`stdout`) and to specified files.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
                              ┌──────────┐
                              ▼          │
►►──ZFILE tee──┬──────┬──────┬────────┬──┴──────────────────────────►◄
               └──-a──┘      └──path──┘
```

-a appends the specified file rather than overwriting it.

path
: is the path name of a file you want to overwrite or append. If multiple path names are specified, the ZFILE tee command will overwrite or append all specified files.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZFILE HELP tee
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.

- The ZFILE tee command is based on a subset of the tee utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

Examples

In the following example, the `stdout` from the ZFILE ls command is piped to the ZFILE tee command and then printed on the screen. The file named `ls.output` is overwritten.

```
User:   ZFILE ls | tee ls.output

System: FILE0001I 08:14:31 START OF DISPLAY FROM ls | tee ls.output
        dev
        etc
        tmp
        usr
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

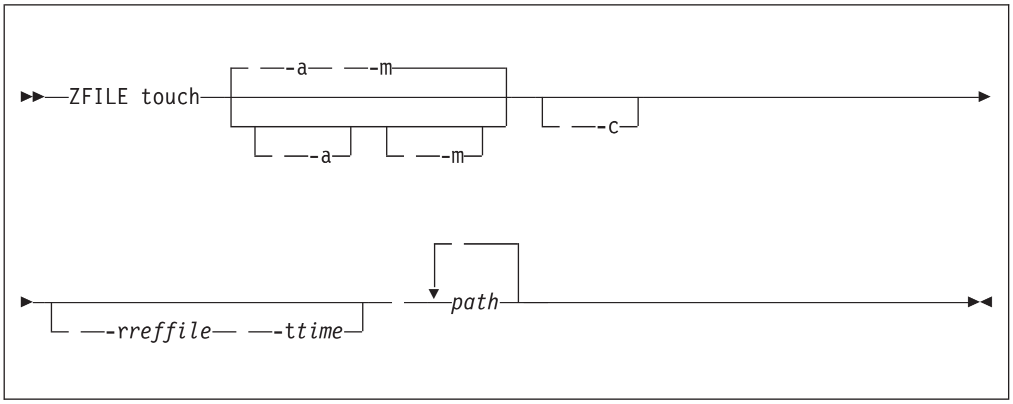
ZFILE touch–Change the Time Stamp of a File

Use this command to change the time stamp of a file or to create a zero-length (empty) file.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format



-a changes the access time, which is the time when the file was last accessed with a read or write operation.

-m changes the modification time, which is the time when the file was last accessed with a write operation.

-c indicates that you do not want to create a file if the file does not exist.

-r*reffile*
specifies a file whose corresponding time is to be used instead of the current time, where *reffile* is the file name.

-t*time*
specifies the time to use instead of the current time. Specify *time* in the format *CCYYMMDDhhmm.SS*, where:

CC
is the first 2 digits of the year (that is, the century). This is optional.

YY
is the last 2 digits of the year. This is optional.

MMDDhhmm
is the month, day, hour, and minutes. You ***must*** specify this portion of the time.

SS
is the seconds. This is optional.

path
is the path name of the file whose time stamp you want to change. If the file does not exist, a new zero-length file is created.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP touch
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE touch -a -c** *path*
  - **ZFILE touch -ac** *path*
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE touch command is based on a subset of the touch utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.
- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

  escape character (\\)
  preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

  ```
  User:   ZFILE echo \$PATH is $PATH

  System: $PATH is /bin:/usr/bin:.
  ```

  single quotation marks (’ ’)
  preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering

`ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (" ")
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (`), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (")	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

In the following example, the access and modification time stamps are changed.

```
User:   ZFILE touch test.file

System: FILE0003I 15:31:22 touch test.... COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE tr–Translate or Delete Characters

Use this command to substitute or delete characters from the standard input (`stdin`) stream and pass the results to the standard output (`stdout`) stream. The results are displayed on the screen, but the input file is not changed.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
►►─ZFILE tr─┬─┬────────────┬─ ─string1─┬───────────┬─┬──────────►◄
            │ ├─ ─-c─ ─-s─┤            └─ ─string2─┘ │
            │ └─ ─-s──────┘                          │
            ├─┬────────────┬─ ─string1───────────────┤
            │ ├─ ─-c─ ─-d─┤                          │
            │ └─ ─-d──────┘                          │
            └─┬────────────────────┬─ ─string1─ ─string2─┘
              ├─ ─-d─ ─-s─────────┤
              └─ ─-c─ ─-d─ ─-s────┘
```

-c complements the set of characters specified in *string1*. The ZFILE tr command constructs a case-sensitive string of characters not contained in *string1* and uses that constructed string as the translation control string.

-s replaces characters in *string1* with the corresponding positional characters in *string2* while eliminating consecutive duplicate characters. If *string2* is not specified, the consecutive duplicate characters are replaced with a single occurrence of itself. The actions of the -s parameter take place after all other deletions and translations.

-d deletes all instances of characters found in *string1*.

string1
is the translation control string, which consists of characters to be converted into an array and replaced or deleted in translation.

string2
is the translation control string, which is used to replace characters specified in *string1*. Unless the -s parameter is specified, characters in *string1* are replaced one-for-one by characters in *string2*.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP tr
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.

- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE tr -d -s** *string1 string2*
  - **ZFILE tr -ds** *string1 string2*
- The following options can be applied to *string1* and *string2*:

  c1–*c2* Represents all the characters from c1 to c2. For example, `a–g` is understood as `abcdefg`.

  *c**n* Represents the character c repeated *n* times. By default, *n* is interpreted as decimal. If n has a leading 0 however, it is interpreted as octal. For example, if you are looking for instances of `oo`, you can specify `oo` or `o*2`.
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  You can redirect the standard output (`stdout`) stream from the display terminal to a file by specifying one of the redirection characters (> or >>) followed by the file name to which you want the output written. The > character writes the output to a file. The >> character appends the output to an existing file.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE tr command is based on a subset of the tr utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.

Examples

In the following example, file `tempfile` contains string `aaaabccccb`. The -s parameter indicates that characters specified in *string1* (`abc`) are replaced by corresponding, positional characters in *string2* (`xyz`) while duplicated characters are eliminated:

```
User:   ZFILE tr -s abc xyz < tempfile

System: FILE0001I 08:14:31 START OF DISPLAY FROM tr -s abc xyz < tempfile
        xyzy
        END OF DISPLAY
```

In the following example, file `tempfile` contains the following single string:

```
one two three
```

Each word is separated by two tab characters (\t). Note that in the display, the words `one`, `two`, and `three` have not been separated by the new-line (\n) character. The actions of the -s parameter take place after all other deletions and translations.

```
User:   ZFILE tr -ds "\t" "\n" < tempfile

System: FILE0001I 08:14:31 START OF DISPLAY FROM tr -ds "\t" "\n" < tempfile
        onetwothree
        END OF DISPLAY
```

In the following example, all lowercase characters from a to s are replaced by their uppercase equivalent. File `tempfile` contains the following:

```
one
two
three
```

Each word is separated with a new-line character (\n).

```
User:   ZFILE tr a-s A-S < tempfile

System: FILE0001I 08:14:31 START OF DISPLAY FROM tr a-s A-S < tempfile
        ONE
        twO
        tHREE
        END OF DISPLAY
```

Related Information

See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.

ZFILE unset–Unset Values of ZFILE Environment Variables

Use this command to remove ZFILE environment variables for the current user.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
                  ┌──────────────┐
                  ▼              │
►►──ZFILE unset───  ──variable───┴──────────────────────────────────►◄
```

variable
is the ZFILE environment variable to remove.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP unset
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- This command does not read from the standard input (`stdin`) stream.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (`2>` or `2>>`) followed by the file name to which you want the error output written. The `2>` character writes the error output to a file. The `2>>` character appends the error output to an existing file.

  Note: When you use the `>` or `2>` character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the `>>` or `2>>` character.
- The ZFILE unset command is based on a subset of the unset utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.

Examples

In the following example, environment variable PATH is removed.

```
User:   ZFILE unset PATH

System: FILE0003I 08:14:31 unset PATH COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY.
```

Related Information

- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See "ZFILE export–Create or Display ZFILE Environment Variables" on page 460 for more information about reserved environment variables.

ZFILE xargs–Construct an Argument List and Run a Command

Use this command to build and run additional commands. The ZFILE xargs command accepts arguments passed from the standard input (`stdin`) stream, completes and runs the command.

Requirements and Restrictions

- The parameters for this command are case-sensitive. You ***must*** enter the parameters exactly as shown in the syntax diagram.
- You can enter this command only in CRAS state or higher.

Format

```
                                         ┌─ ──-n── ──1──────┐
►►──ZFILE xargs──┬───────┬──┬───────┬──┼──────────────────┼─────────────────────►
                 └─ ──-t─┘  └─ ──-x─┘  └─ ──-n── ──number─┘

                          ┌─ ──echo────────────────────────┐
►──┬──────────────┬──┼──────────────────────────────────┼──────────────────────►◄
   └─ ──-s── ──size─┘  │              ┌─────────────────┐ │
                          └─ ──funcmsg──▼─┬─────────────┬─┴─┘
                                          └─ ──argument─┘
```

-n *number*
: specifies the number of arguments processed at a time from `stdin` by the ZFILE xargs command, where *number* is a whole number greater than 0. For example:

```
zfile find . -name "*.c" | xargs -n 2 echo
```

takes two arguments at a time from the results of the ZFILE find command and passes them to the ZFILE echo command, or `echo`. Once the ZFILE echo command is run, the process repeats until there are no more arguments in `stdin`.

-t
: writes the contents of `stdin` to standard error (`stderr`) before the arguments are processed by the command specified in the *funcmsg* parameter. This allows you to see the original information that was passed through the ZFILE xargs command to *funcmsg*.

-x
: ends the ZFILE xargs command if `stdin` is larger than the number of bytes specified by -s *size*. If the -s parameter is not specified, the -x parameter will be activated when `stdin` is larger than system variable LINE_MAX, which is currently defined as 2048 bytes.

-s *size*
: sets the maximum number of bytes of the argument list, where *size* is a whole number greater than 0 but less than or equal to system variable LINE_MAX, which is currently defined as 2048 bytes.

funcmsg
: specifies the command to run using the information in `stdin`. Do not type the

`ZFILE` portion of the command again, but rather just the keyword such as `echo` or `grep`. If no command is specified, `stdin` is passed to the ZFILE echo command.

argument
specifies a parameter needed to run the operation indicated in *funcmsg*.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFILE HELP xargs
  ZFILE HELP
  ZFILE ?

  In addition, the correct command syntax is displayed if you enter the syntax incorrectly.
- The LINE_MAX system variable is defined in the `syslimit.h` header file.
- If you specify more than one parameter, you can specify these parameters separately or together. For example, you can enter one of the following:
  - **ZFILE find . -name ″*.t″ | xargs -t -n3**
  - **ZFILE find . -name ″*.t″ | xargs -tn3**
- You can use a vertical bar, or pipe (|), to direct data so the output from one process becomes the input to another process. This type of one-way communication allows you to combine ZFILE commands on one line to create a pipeline. For example, the following pipeline uses the standard output (`stdout`) stream from the ZFILE ls command and redirects it to the standard input (`stdin`) stream of the ZFILE grep command to search for those lines containing the word Jan.

  ```
  ZFILE ls -l | grep Jan
  ```

  The result is filtered output from the ZFILE ls command displaying only the lines containing the word Jan in any position.

  You can use pipes only with a combination of ZFILE commands where the command on the left-hand side of the pipe provides data through `stdout` and the right-hand side accepts data through `stdin`.
- You can redirect the standard input (`stdin`) stream from the keyboard to a file by specifying the redirection character (<) followed by the file name from which you want the input read.

  This command does not write to the standard output (`stdout`) stream.

  You can redirect the standard error (`stderr`) stream from the display terminal to a file by specifying one of the redirection characters (2> or 2>>) followed by the file name to which you want the error output written. The 2> character writes the error output to a file. The 2>> character appends the error output to an existing file.

  Note: When you use the > or 2> character, if the file that you are redirecting data to already exists, the file is overwritten and any data in that file is lost. If you do not want to overwrite the file, ensure that you use the >> or 2>> character.
- The ZFILE xargs command is based on a subset of the xargs utility of the Portable Operating System Interface for Computer Environments (POSIX) standards.

- You can use the following wildcard characters in the path name to select files that satisfy a particular name pattern:

Wildcard	Description
*****	Matches any string, including the empty string.
?	Matches any single character.
[..]	Matches any one of the characters between the brackets. You can specify a range of characters by separating a pair of characters with a dash (-).

- This command supports the following three quoting mechanisms, which allow you to override the special meaning of some characters:

escape character (\)
preserves the literal value of the character that follows. To ignore the special meaning of a character, escape it by placing a backslash (\) in front of it. In the example that follows, environment variable PATH is `/bin:/usr/bin:.`, the first dollar sign ($) is escaped because of the preceding backslash, and the second dollar sign takes on a special meaning.

```
User:   ZFILE echo \$PATH is $PATH

System: $PATH is /bin:/usr/bin:.
```

single quotation marks (' ')
preserves the literal value of all characters between the opening single quotation mark and the closing single quotation mark. For example, entering `ZFILE echo '*'` displays an asterisk character (*). Without the single quotation marks, the files in the current working directory are displayed.

double quotation marks (″ ″)
preserves the literal value of all characters between the opening double quotation mark and the closing double quotation mark except the dollar sign ($), the backquote (‘), and the backslash (\). This allows you to use the escape character inside double quotation marks, which you cannot do in single quotation marks.

For example, entering `ZFILE echo "\$PATH is \"$PATH\""` displays `$PATH is "/bin:/usr/bin:."`

If you want to use the literal meaning of any of the following characters, you must always use a quoting mechanism:

left angle bracket (<)	right angle bracket (>)	ampersand (&)
backquote (`)	backslash (\)	dollar sign ($)
double quotation mark (″)	new-line (\n)	left parenthesis (( )
right parenthesis ( ))	semicolon (;)	single quotation mark (')
blank space	tab	vertical bar (|)

In addition, use a quoting mechanism when any of the following characters are used in a way that takes on a special meaning but you want to use the literal meaning:

asterisk (*)	equal sign (=)	left square bracket ([ )
number sign (#)	question mark (?)	tilde (~)

Examples

The following example first writes the results of the ZFILE find command to `stderr` and then searches those files by using the ZFILE grep command to locate files that contain string `rmdir`.

```
User:   ZFILE find . -name '*.t' | xargs -t grep rmdir

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -name '*.t' | xargs -t ...
        ./test.t: rmdir 'tmp'
        ./xyz.t: rmdir 'temp'
        END OF DISPLAY

System: FILE0001I 08:14:31 START OF ERROR DISPLAY FROM find . -name '*.t'| xargs -t ...
        grep echo ./abc.t ./new.t ./test.t ./xyz.t
        END OF DISPLAY
```

The following example uses three columns to display the file names that end in `.c` in or below the current directory.

```
User:   ZFILE find . -name "*.c" | xargs -n 3 echo

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -name "*.c" | xargs -n 3 echo
        ./file1.c  ./file2.c  ./file3.c
        ./file4.c  ./file5.c  ./file6.c
        END OF DISPLAY
```

The following example uses the -s parameter to set the maximum argument list size to 50 bytes and then displays all files that end in `.c` in or below the current directory.

```
User:   ZFILE find . -name "*.c" | xargs -s50

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -name "*.c" | xargs -s50
        ./file1.c  ./file2.c  ./file3.c  ./file4.c
        ./file5.c  ./file6.c
        END OF DISPLAY
```

The following example uses the -s parameter to set the maximum argument list size to 50 bytes. The -x parameter limits the size of the input from the ZFILE find command to 50 bytes. The input is larger than 50 bytes so the ZFILE xargs command exits.

```
User:   ZFILE find . -name "*.c" | xargs -s50 -x

System: FILE0001I 08:14:31 START OF DISPLAY FROM find . -name "*.c" | xargs -s...
        xargs: insufficient space for arguments
        END OF DISPLAY
```

Related Information

- See *Information Technology—Portable Operating System Interface for Computer Environments (POSIX)* for more information about the POSIX standards.
- See “ZFILE echo–Write Data to the Standard Output Stream” on page 457 for more information about removing files.
- See “ZFILE find–Find a File” on page 463 for more information about removing files.

ZFILT–Display or Refresh TCP/IP Packet Filtering Rules

Use this command to display or refresh TCP/IP packet filtering rules that are defined for your TPF system.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZFILT── ──┬─DISplay─┬──────────────────────────────────────────────────►◄
              └─REFresh─┘
```

DISplay
displays the packet filtering rules and the number of packets that have applied to each rule.

REFresh
refreshes the core copy of the packet filtering rules from the current copy of the `/etc/iprules.txt` file.

Notes:

1. Changes to the packet filtering rules take effect immediately after you enter ZFILT REFRESH.
2. If you specify the REFRESH parameter and the same rule exists in both the previous and updated versions of the `/etc/iprules.txt` file, the number of packets that applied to that rule are not reset to 0 across the refresh operation; therefore, you will not lose any previous data related to that rule. For example, assume a rule existed in the previous version of the file and there were 500 packets for which this rule applied. If the same rule exists in the updated file after you enter ZFILT REFRESH, the number of packets for which the rule applies will still be 500.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZFILT HELP
ZFILT ?

Examples

In the following example, the current packet filtering rules are displayed, where:

RULE
is the rule number. The last rule, DEF, is the default action.

ACTION
is the action that will be taken if the rule is applied to an input packet.

REMOTE NETWORK
is the IP network that sent the packet to the TPF system.

PORT
is the port number of the TPF application.

PROTO
is the protocol.

ICMPTYPE
is the type of Internet Control Message Protocol (ICMP) message.

PACKETS
is the number of packets for which this rule has been applied.

Note: For TCP packets, the rules are examined only for connection requests.

```
User:   ZFILT DISPLAY

System: FILT0001I 17.20.24 DISPLAY PACKET FILTERING RULES

        RULE   ACTION     REMOTE NETWORK     PORT   PROTO  ICMPTYPE   PACKETS
        ----   ------   ------------------  -----   -----  -------- ----------
           1    ALLOW       9.117.121.0/24   5001     TCP                  671
           2   REJECT                        5001     TCP                   23
           3    ALLOW         1.123.0.0/16     25     TCP                 2134
           4    ALLOW         1.123.0.0/16   6666     UDP                  430
           5     DENY          1.56.0.0/16                                  22
           6     DENY          1.56.0.0/16           ICMP         8        323
           7    ALLOW      9.117.121.35/32   5002     TCP                  442
           8   REJECT                        5002     TCP                 1230
         DEF    ALLOW                                                   333523

        END OF DISPLAY+
```

Related Information

See the *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP packet filtering firewall support.

ZFINT–Schedule Initialization of the File System

Use this command to do any of the following:

- Schedule the initialization of the file system during the next cycle to NORM state.
- Request status of the file system.
- Add additional #INODE and #FLOCK fixed file records to the file system.
- Determine the current usage of #INODE and #FLOCK fixed file records.
- Set the number of entries for the file system directory and i-node caches.

Requirements and Restrictions

- If you are initializing both file system support and TPF collection support ***at the same time***, you must enter the ZOODB INIT command, which makes it unnecessary for you to enter the ZFINT command.
- If you enter the ZOODB INIT command when some subsystems are not in NORM state, you must enter the ZFINT command at the next cycle to NORM state to initialize file system support in those subsystems.
- Cache support is present only when the file system has been initialized in the basic subsystem (BSS).

Format

```
►►──ZFINT── ──┬─ON──┬───────┬─┬──────────────────────────────────────────►◄
              │     └─ ──BP─┘ │
              ├─OFF───────────┤
              ├─STATUS────────┤
              ├─ADD── ──num───┤
              ├─Usage─────────┤
              ├─DCache dcachenum─┤
              └─ICache icachenum─┘
```

ON
forces the file system to be initialized during the next cycle up to NORM state if the file system has not been initialized already. If you IPL the system before going to NORM state, you must enter the ZFINT command with the ON parameter again.

BP
forces the file system to be reinitialized during the next cycle up to NORM state.

Attention: Use the BP option with caution to reinitialize the file system. After you initialize the file system you normally should ***not*** reinitialize the file system. The BP option will delete any existing files in the file system without cleaning up any system resources that are attached to those files. Other TPF processors should not use the file system while you are reinitializing the file system.

OFF
specifies that you do not want to initialize the file system during the next cycle up to NORM state. The OFF parameter cancels the ZFINT command with the ON parameter.

STATUS
displays the status of the file system initialization indicator.

ADD *num*
adds to the file system the specified number of #INODE and #FLOCK fixed file records that have already been added to the TPF file address compute program (FACE) table, where *num* is a positive decimal number from 1–999999.

Use the ADD parameter to increase the number of #INODEand #FLOCK fixed file records in the file system. You must ensure that the FACE tables are compatible on all active TPF images.

Usage
tells how many #INODE and #FLOCK fixed file records are initialized and can be used by the TPF system. This parameter also displays the current cache sizes from the #IZERO fixed file record if the command has been entered on the BSS. The current cache sizes that are displayed may not be the actual cache size if the size has been changed since the last IPL. The actual size of the allocated cache cannot be changed unless one of the following occurs:

- The TPF system is IPLed.
- The cache is deleted and you enter the ZFINT command with a new cache size.

DCache *dcachenum*
allocates the number of entries for the file system directory cache, where *dcachenum* is a decimal number in the range 0–999 999. If you specify 0, the corresponding cache will not be allocated or, if it is currently active, it will be released. This function is valid only when entered on the BSS.

ICache *icachenum*
allocates the number of entries for the file system i-node cache, where *icachenum* is a decimal number in the range 0–999 999. If you specify 0, the corresponding cache will not be allocated or, if it is currently active, it will be released. This function is valid only when entered on the BSS.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFINT HELP
  ZFINT ?
- When you enter the ZFINT command with the ON parameter specified and then IPL the TPF system again before the file system is initialized, you must enter the ZFINT command again with the ON parameter specified.

Examples

In the following example, the file system will be initialized when the TPF system reaches NORM state.

```
User:   ZFINT ON

System: FINT0001I 06.04.02 FILE SYSTEM INITIALIZATION PENDING
```

In the following example, the file system will not be initialized when the TPF system reaches NORM state.

```
User:   ZFINT OFF

System: FINT0002I 07.14.01 FILE SYSTEM INITIALIZATION NOT PENDING
```

The following example requests the status of the file system initialization indicator. The response indicates that the file system will not be initialized when the TPF system reaches NORM state.

```
User:   ZFINT STATUS

System: FINT0002I 08.21.03 FILE SYSTEM INITIALIZATION NOT PENDING
```

In the following example, the number of #INODE and #IZERO fixed file records that are initialized is requested.

```
User:   ZFINT USAGE

System: FINT0017I 18.36.46 FILE SYSTEM CACHE SIZE:
                               DIRECTORY CACHE 1797  INODE CACHE  1347

        FINT0011I 18.36.46 490 OF 499 INODE AND 10 OF 10 IZERO RECORDS ARE IN USE
```

In the following example, no #INODE or #IZERO fixed file records are initialized; therefore, the file system is not initialized.

```
User:   ZFINT USAGE

System: FINT0012I 14.14.53 FILE SYSTEM NOT INITIALIZED
```

In the following example, the file system directory cache is created and set to hold 200 000 entries. The #IZERO fixed file record will also be updated with the new value.

```
User:   ZFINT DCache 200000

System: FINT0015I 14.14.53 FILE SYSTEM DIRECTORY CACHE CREATED
```

Related Information

- See *TPF Concepts and Structures* and the *TPF C/C++ Language Support User's Guide* for more information about TPF file system support.
- See *TPF System Generation* for more information about using the CORREQ macro to determine the amount of frames you need to define for the system heap.

ZFKPA–Reply to Insufficient Core Message

Use this command to specify whether restart should continue with reduced core or end when keypoint M (MK0CK) indicates that there is not enough core.

Requirements and Restrictions

You can enter this command only when you are prompted by the TPF system.

Format

```
►►──ZFKPA── ──┬─RED─┬──────────────────────────────────────────────►◄
              └─CAN─┘
```

RED
continues the initial program load (IPL) with the reduced block counts.

CAN
cancels the IPL by exiting the restart ECB.

Additional Information

None.

Examples

The IPL is continued in the following example.

```
System: FKPA0001A 08.40.01
        INSUFFICIENT CORE FOR WORKING STORAGE
        TOTAL CORE AVAILABLE = 006D6E80
        TOTAL CORE REQUIRED  = 009AF1A0
        FKPA0007A 08.40.01
        SPECIFY OPTION-
        TO CONTINUE RESTART ENTER -
           ZFKPA RED
        TO CANCEL RESTART ENTER -
           ZFKPA CAN

User:   ZFKPA RED

System: FKPA0008A 12.55.01 NUMBER OF COMMON BLOCKS REDUCED TO 3274
```

Related Information

See *TPF Main Supervisor Reference* for more information about keypoint M.

ZFMNT–Mount General File

Use this command to make a general file data set available to the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZFMNT── ──number── ──address──┬──────┬──────────────────────────────────►◄
                                  └─ ──BP─┘
```

number
is a data set number from 0–99.

address
is a device address.

BP
mounts the specified data set without verifying the label first.

Additional Information

None.

Examples

In the following example, the specified data set is mounted on device 04E7.

```
User:   ZFMNT 03 04E7

System: FMNT0001I 07.10.32 GENERAL FILE DATA SET  3 MOUNTED ON 04E7
```

In the following example, the specified data set is mounted without verifying the label first.

```
User:   ZFMNT 03 04E7 BP

System: FMNT0001I 07.10.32 GENERAL FILE DATA SET  3 MOUNTED ON 04E7
        FMNT0001I 07.10.32 DATA SET  3 MAY/MAYNOT BE CONTAINED ON GENERAL FILE
                           MOUNTED ON 04E7
```

Related Information

See *TPF Database Reference* for more information about general file support.

ZFMSG–Process Command Characteristics

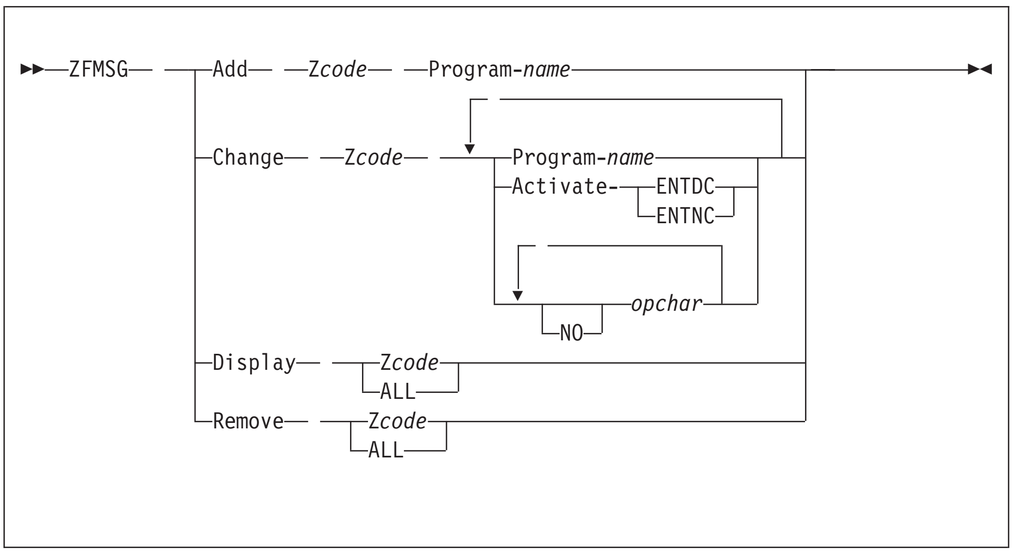
Use this command to do the following:
- Create a new user-defined command
- Change the operating characteristics of a command
- Display the operating characteristics of a command
- Delete a user-defined command
- Delete any changes that were made to a command that was provided by the TPF system.

A *user-defined* command is a command that you define and create. It is not provided by the TPF system.

Requirements and Restrictions

You can enter this command from any subsystem only if the multiple database function (MDBF) is generated.

Format



Add
creates a new user-defined command.

Note: After you create a new user-defined command, use the CHANGE parameter to define the operating characteristics.

Change
changes the operating characteristics of a command.

Display
displays the operating characteristics of a command.

Remove
deletes a user-defined command or deletes the changes that were made to a command that was provided by the TPF system.

Z*code*
is a command, where *code* is a 1- to 4-character alphanumeric secondary action code (SAC).

Note: If you do not specify a secondary action code, a blank character is assumed.

Program-*name*
is the 4-character name of the service program to be started by the new command.

Activate-
specifies the activation type to be used when passing control to the service program.

ENTDC
specifies an activation type of ENTDC.

ENTNC
specifies an activation type of ENTNC.

ALL
displays all of the commands that were created or changed, deletes all of the user-defined commands, or deletes all of the changes that were made to commands that are provided by the TPF system.

NO
turns off the following operating characteristics.

opchar
is one or more of the following operating characteristics:

For Command Indicators:

EP
The command is valid only in the EP processor.

BSS
The command is valid only in the basic subsystem.

RBS
Send the command to the basic subsystem for processing.

SSU
Use the DBI of the first subsystem user when processing the command.

RST
The command is restricted to authorized CRAS terminals.

DRM
Allows the command to be processed in a dormant subsystem user.

ALL
The command is allowed from any terminal.

TST
The command is allowed only when the system is operating in test mode.

PCO
The command is allowed only from prime CRAS.

For Functional Support Consoles:

RO
Send a copy of the response to RO CRAS.

PRC
Send a copy of the response to prime CRAS.

TAPE
Send a copy of the response to the tape console.

DASD
Send a copy of the response to the DASD console.

COMM
Send a copy of the response to the communications console.

AUDT
Send a copy of the response to the audit trail console.

RDBS
Send a copy of the response to the real-time database services (RDBS) console.

For Additional Indicators:

STA
The command is processed using the SNA type A format, which allows you to enter the command without parameters. This parameter is identical to the NOSTB parameter.

STB
The command is processed using the SNA type B format, which requires you to enter the command with parameters. This parameter is identical to the NOSTA parameter.

USR
The command is a user-defined command.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZFMSG HELP
  ZFMSG ?
- Changes made with this command take effect immediately.
- No conflict checking is performed when you change command indicators. For example, ALL and PCO can be specified at the same time but are in direct conflict with each other. Normally, a higher authorization overrides a lower one. In this example, the command is allowed only from the prime CRAS. The ALL parameter is ignored.

Examples

The ZANEW command is created in the following example. When you enter this command, the CVXT program is started.

```
User:   ZFMSG ADD ZANEW PROGRAM-CVXT

System: FMSG0002I ZANEW ADDED
        ZANEW ACTIVATES CVXT VIA ENTDC
        NO COMMAND INDICATORS SET
        FUNCTIONAL SUPPORT CONSOLES SET
         RO   - RO CRAS
         AUDT - AUDIT TRAIL CONSOLE
        NO ADDITIONAL INDICATORS SET
```

The operating characteristics of the ZANEW command are changed in the following example.

```
User:   ZFMSG CHANGE ZANEW RBS DRM NOAUDT

System: FMSG0004I ZANEW CHANGED
        ZANEW ACTIVATES CVXT VIA ENTDC
        COMMAND INDICATORS SET
         RBS - ROUTE TO BASIC SUBSYSTEM
         DRM - ALLOW IN DORMANT SS USER
        FUNCTIONAL SUPPORT CONSOLES SET
         RO   - RO CRAS
        NO ADDITIONAL INDICATORS SET
```

The ZANEW command is deleted in the following example.

```
User:   ZFMSG REMOVE ZANEW

System: FMSG0003I ZANEW REMOVED
```

The operating characteristics defined for the ZNETW command are displayed in the following example.

```
User:   ZFMSG DISPLAY ZNETW

System: FMSG0001I 07.10.37 BEGIN DISPLAY FOR ZNETW
        ZNETW ACTIVATES CSEA VIA ENTDC
        COMMAND INDICATORS SET
         BSS - BASIC SUBSYSTEM ONLY
         SSU - USE FIRST SUBSYSTEM USER DBI
        FUNCTIONAL SUPPORT CONSOLES SET
         COMM - COMMUNICATIONS CONSOLE
         AUDT - AUDIT TRAIL CONSOLE
        ADDITIONAL INDICATORS SET
         STB - USE SNA TYPE 'B' FORMAT
```

Related Information

See “User-Defined Commands” on page 16 for more information about user-defined commands.

ZFRST ABORT–Abort the Restore Function

Use this command to immediately end the restore function, stop restoring exception records, or stop restoring record logs. You can also use this command to temporarily stop restoring one or more online modules (physical DASD units).

Requirements and Restrictions

- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format

```
►►──ZFRST ABORT── ──┬─ALL───────────────┬──────────────────►◄
                    │         ┌─/─────┐ │
                    └─MOD── ──▼─module─┴─┘
```

ALL
: ends the restore function, stops restoring exception records, or stops restoring record logs.

MOD
: temporarily stops restoring one or more specific online modules.

module
: is a 3-digit hexadecimal symbolic module number.

Additional Information

If you temporarily stop restoring one or more specific online modules, those modules are automatically restored when you remount the restore tape.

Examples

The restore function is ended for all modules in the following example.

```
User:   ZFRST ABORT ALL

System: FRST0042I 14.42.04 ABORT IN PROGRESS
        FRST0016I 14.42.05 MOD 01E ON 0520 FROM TAPE ON 423 - ABORTED
        FRST0016I 14.42.05 MOD 01F ON 0521 FROM TAPE ON 480 - ABORTED
        FRST0015I 14.42.05 ABORTED
        COTC0087A 14.42.06 TCLS WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243  NOBLK
        COTC0087A 14.42.06 TCLS WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248  NOBLK
```

The restore function temporarily stops restoring modules 01E and 01F in the following example.

ZFRST ABORT

```
User:   ZFRST ABORT MOD 01E/01F

System: FRST0016I 14.44.58 MOD 01E ON 0520 FROM TAPE ON 423 - ABORTED
        FRST0016I 14.44.58 MOD 01F ON 0521 FROM TAPE ON 480 - ABORTED
        COTC0087A 14.44.58 TCLS WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243  NOBLK
        COTC0087A 14.44.58 TCLS WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248  NOBLK
```

Related Information

See *TPF Database Reference* for more information about the restore function, exception recording, and record logging.

ZFRST ALTER–Change the Number of Simultaneous Restores Allowed

Use this command to change the number of restores allowed at one time on the current processor.

Requirements and Restrictions

The restore function must be running.

Format

```
►►──ZFRST ALTER── ──ECB── ──LEVEL── ──max──────────────────────────────►◄
```

max
is a decimal number from 1–32 that specifies the maximum number of restores allowed at one time.

Additional Information

- Enter the ZFRST STATUS command to display the current maximum number of restores that are allowed at one time.
- The ZFCAP CLEAR and ZFRST CLEAR commands reset the maximum number of restores to the default value defined in the master copy of the capture and restore working keypoint.

Examples

The maximum number of restores allowed at one time is changed in the following example.

```
User:   ZFRST ALTER ECB LEVEL 16

System: FRST0084I 15.37.21 ECB LEVEL ALTERED TO 16
```

Related Information

See *TPF Database Reference* for more information about the restore function.

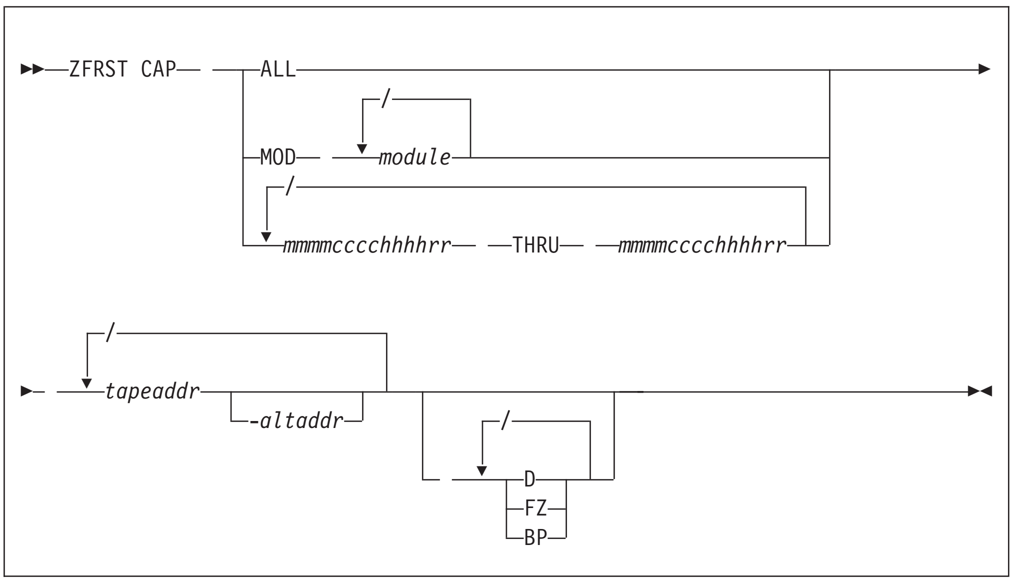
ZFRST CAP–Start the Restore Function

Use this command to start the restore function. You can restore all the online modules (physical DASD units), only specific online modules, or a specific address range.

Requirements and Restrictions

- Specify at least one tape device for each participating processor.
- Load the restore tapes (BXA–BXZ, BX0–BX5) and ensure that they are ready to use.
- Each participating processor must be in UTIL state or higher.
- An entry for the restore function (RST*x*, where *x* is an identifier that is unique for each processor) must exist and be active in the processor resource table (PROT) for each participating processor. Use the ZPROT command to define the PROT entry.
- Do not use the ALL prefix with this command.

Format



ALL
restores all the online modules.

MOD
restores one or more specific online modules.

module
is a 3-digit hexadecimal symbolic module number.

mmmmccccchhhhrr **THRU** *mmmmccccchhhhrr*
restores a specific address range, where *mmmmccccchhhhrr* is a 14-digit (7-byte) hexadecimal file address:

mmmm
is the 4-digit hexadecimal symbolic module number.

cccc
is the 4-digit hexadecimal cylinder number.

hhhh
is the 4-digit hexadecimal head number.

rr is the 2-digit hexadecimal record number.

tapeaddr
is the 3-digit hexadecimal address of a primary tape device.

altaddr
is the 3-digit hexadecimal address of the alternate tape device. When specified, the restore function can alternate between the primary and alternate tape device during the restore.

Note: The addresses of the primary and alternate tape devices must be on the same device type.

D restores duplicate records.

FZ files zeroed records.

BP
bypasses the reel sequence check.

Additional Information

None.

Examples

The restore function is started for all online modules in the following example. Data is restored from 2 sets of primary and alternate tape devices.

```
User:   ZFRST CAP ALL 423-424/480-481 FZ/BP

System: FRST0119I 11.43.30 DUAL TAPE 423-424 ADDED
        FRST0119I 11.43.30 DUAL TAPE 480-481 ADDED
        FRST0113I 11.43.30 FILE ZEROED RECORD OPTION SELECTED
        FRST0109I 11.43.30 BYPASS REEL SEQUENCE CHECK SPECIFIED
        FRST0010I 11.43.30 STARTED
        COTM0046I 11.43.30 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK
        COTM0046I 11.43.31 TMNT WP1    TAPE BXB MOUNTED ON DEVICE 480
                           VSN A00248 G     S     D38K   SL  NOBLK
        FRST0008I 11.43.31 MOD 01E ON 0520 FROM TAPE ON 423 - STARTED
        FRST0008I 11.43.31 MOD 01F ON 0521 FROM TAPE ON 480 - STARTED
        FRST0121A 11.47.33 TAPE ON 481 FOR MOD 01F - SWITCHED, REMOVE PRIOR
        REEL 480
        COTM0046I 11.47.33 TMNT WP1    TAPE BXB MOUNTED ON DEVICE 481
                           VSN A00249 G     S     D38K   SL  NOBLK
        FRST0121A 11.47.33 TAPE ON 424 FOR MOD 01E - SWITCHED, REMOVE PRIOR
        REEL 423
        COTM0046I 11.47.34 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 424
                           VSN A00245 G     S     D38K   SL  NOBLK
        COTT0382I 11.47.34 TPSW WP1    TAPE BXB SWITCHED FROM 480 TO 481
                           VSN IS NOW A00249
        COTT0087A 11.47.34 TREV WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248  NOBLK
        COTT0382I 11.47.34 TPSW WP1    TAPE BXA SWITCHED FROM 423 TO 424
                           VSN IS NOW A00245
        COTT0087A 11.47.34 TREV WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243  NOBLK
        FRST0009I 11.48.46 MOD 01E ON 0520 FROM TAPE ON 424 - COMPLETED
        FRST0009I 11.48.46 MOD 01F ON 0521 FROM TAPE ON 481 - COMPLETED
        COTC0087A 11.48.46 TCLS WP1    REMOVE BXA FROM DEVICE 424
                           VSN A00245  NOBLK
        COTC0087A 11.48.46 TCLS WP1    REMOVE BXB FROM DEVICE 481
                           VSN A00249  NOBLK
        FRST0134I 11.48.46 PROCESSOR B COMPLETED
        FRST0001I 11.48.46
        RESTORE STARTED 06MAY 1143
        MODULES TO DO        2
        MODULES COMPLETED    2
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE
        FRST0011I 11.48.46 COMPLETED
        DISK ERRORS
        TRKS 000
        RCDS 000
        TAPE ERRORS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   424   423   00              B
          BXB   481   480   00              B
        END OF STATUS DISPLAY
```

The restore function is started for modules 01E and 01F in the following example. Data is restored from 2 sets of primary and alternate tape devices.

```
User:   ZFRST CAP MOD 01E/01F 423-424/480-481 BP

System: FRST0057I 10.54.49 MOD 01E ADDED
        FRST0057I 10.54.49 MOD 01F ADDED
        FRST0119I 10.54.49 DUAL TAPE 423-424 ADDED
        FRST0119I 10.54.49 DUAL TAPE 480-481 ADDED
        FRST0010I 10.54.49 STARTED
        COTM0046I 10.54.50 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 423
                           VSN A00177 G     S     D38K   SL  NOBLK
        COTM0046I 10.54.50 TMNT WP1    TAPE BXB MOUNTED ON DEVICE 480
                           VSN A00180 G     S     D38K   SL  NOBLK
        FRST0008I 10.54.51 MOD 01E ON 0520 FROM TAPE ON 423 - STARTED
        FRST0008I 10.54.51 MOD 01F ON 0521 FROM TAPE ON 480 - STARTED
        FRST0121A 10.57.29 TAPE ON 424 FOR MOD 01E - SWITCHED, REMOVE PRIOR REEL FROM 423
        COTM0046I 10.57.29 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 424
                           VSN A00178 G     S     D38K   SL  NOBLK
        FRST0121A 10.57.29 TAPE ON 481 FOR MOD 01F - SWITCHED, REMOVE PRIOR REEL FROM 480
        COTM0046I 10.57.29 TMNT WP1    TAPE BXB MOUNTED ON DEVICE 481
                           VSN A00181 G     S     D38K   SL  NOBLK
        COTT0382I 10.57.29 TPSW WP1    TAPE BXA SWITCHED FROM 423 TO 424
                           VSN IS NOW A00178
        COTT0087A 10.57.29 TREV WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00177  NOBLK
        COTT0382I 10.57.29 TPSW WP1    TAPE BXB SWITCHED FROM 480 TO 481
                           VSN IS NOW A00181
        COTT0087A 10.57.29 TREV WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00180  NOBLK
        FRST0009I 10.58.08 MOD 01F ON 0521 FROM TAPE ON 481 - COMPLETED
        COTC0087A 10.58.09 TCLS WP1    REMOVE BXB FROM DEVICE 481
                           VSN A00181  NOBLK
        FRST0009I 10.58.11 MOD 01E ON 0520 FROM TAPE ON 424 - COMPLETED
        FRST0134I 10.58.11 PROCESSOR B COMPLETED
        COTC0087A 10.58.11 TCLS WP1    REMOVE BXA FROM DEVICE 424
                           VSN A00178  NOBLK
        FRST0001I 10.58.11
        RESTORE STARTED 05AUG 1054
        MODULES TO DO        2
        MODULES COMPLETED    2
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   032
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE
        FRST0011I 10.58.11 COMPLETED
        DISK ERRORS
        TRKS 000
        RCDS 000
        TAPE ERRORS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   424   423   00               B
          BXB   480   481   C1               B
        END OF STATUS DISPLAY
```

The restore function is started for a specific address range in the following example. Data is restored from a primary and an alternate tape device.

```
User:   ZFRST CAP 001E0047000301 THRU 001E004700040A 423-424

System: FRST0119I 16.15.52 DUAL TAPE 423-424 ADDED
        FRST0010I 16.15.52 STARTED
        COTM0046I 16.15.52 TMNT WP1    TAPE BXA MOUNTED ON DEVICE 423
                           VSN A00243 G     S     D38K   SL  NOBLK
        FRST0008I 16.15.52 MOD 01E ON 0520 FROM TAPE ON 423 - STARTED
        FRST0009I 16.16.21 MOD 01E ON 0520 FROM TAPE ON 423 - COMPLETED
        FRST0134I 16.16.21 PROCESSOR B COMPLETED
        COTC0087A 16.16.21 TCLS WP1    REMOVE BXA FROM DEVICE 423
                           VSN A00243  NOBLK
        FRST0001I 16.16.21
        RESTORE STARTED 06MAY 1616
        MODULES TO DO        1
        MODULES COMPLETED    1
        MODULES REMAINING    0
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   016
        CURRENT ACTIVE ECB    000
        XCP RECORDING ACTIVE
        DISK ERRORS
        TRKS 000
        RCDS 000
        TAPE ERRORS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                            KPT  /  CP
          BXA   423   424   00              B
        END OF STATUS DISPLAY
        FRST0011I 16.16.21 COMPLETED
```

Related Information

See *TPF Database Reference* for more information about the restore function.

ZFRST CLEAR–Initialize the Capture and Restore Working Keypoint

Use this command to initialize the capture and restore working keypoint. The fields in the keypoint are reset to the default values stored in the master copy of the keypoint.

Requirements and Restrictions

If you enter the ZFRST CLEAR command with the BP parameter while the restore function is running, you must enter the ZTDEV RESUME command for each tape device that was previously included in the capture and restore working keypoint.

Format

```
►►──ZFRST CLEAR──┬───────┬──────────────────────────────────────────────►◄
                 ├─ALL─┤
                 └─BP──┘
```

ALL
initializes the capture and restore working keypoint on all active processors if the restore function is not running. If you do not specify the ALL parameter, the capture and restore working keypoint is initialized only on the current processor.

BP
initializes the capture and restore working keypoint on the current processor when the restore function is running.

Attention: Initializing the capture and restore working keypoint when the restore function is running causes unexpected results. Use this parameter only when other attempts to end the restore function fail.

Additional Information

- This command resets the following values:
  - Capture maximums that were defined using the ZFCAP CHANGE command
  - Maximum number of captures allowed at one time that were defined using the ZFCAP ALTER command
  - Input/output (I/O) delay time factor that was defined using the ZFCAP IOTIME command.
  - Maximum number of restores allowed at one time that were defined using the ZFRST ALTER command.
- You can also enter the ZFCAP CLEAR command to initialize the capture and restore working keypoint.

Examples

The capture and restore working keypoint is initialized on the current processor in the following example.

```
User:   ZFRST CLEAR

System: FRST0060I 13.36.10 FCAP/FRST KEYPOINT REINITIALIZED
```

Related Information

See *TPF Database Reference* for more information about the capture and restore working keypoint.

ZFRST KPT–Restore Keypoint Records

Use this command to restore the system keypoint records from the keypoint capture (KPC) tape.

Requirements and Restrictions

- Use the ZTMNT command to mount the KPC tape for input.
- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format

```
►►──ZFRST KPT──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The system keypoint records are restored in the following example.

```
User:   ZFRST KPT

System: COSK0079A 10.31.32 *CP* WP1    MOUNT KPC TAPE FOR INPUT
        WP/ZTMNT KPC 425 AI
        COTM0046I 12.31.38 TMNT WP1    TAPE KPC MOUNTED ON DEVICE 425
                           VSN A00246 G0003 S0001 D38K   SL  NOBLK
        FRST0010I 12.31.38 STARTED
        FRST0089A 12.31.41 MUST IPL SS IN PROCESSOR(S) TO CONTINUE
        FRST0027I 12.31.41 KPT COMPLETE
        COTC0087A 12.31.41 TCLS WP1    REMOVE KPC FROM DEVICE 425
                           VSN A00246  NOBLK
```

Related Information

See *TPF Database Reference* for more information about restoring keypoint records.

ZFRST LOG–Restore Logging Records

Use this command to restore logging records for a range of time. You can restore logging records for all the online modules (physical DASD units), for specific online modules, or for a specific address range.

Requirements and Restrictions

- Use the ZTMNT command to mount the logging tape (XAT–XAY) for input. Specify the BP parameter.

  If more than one processor was active when the logging records were captured, you must mount the logging tapes as follows:
  - Mount the logging tapes (RTT–RTY) for processor 1 as XAT–XAY, respectively.
  - Mount the logging tapes (RTT–RTY) for processor 2 as XBT–XBY, respectively.
  - Mount the logging tapes (RTT–RTY) for processor 3 as XCT–XCY, respectively, and so on.
- Each participating processor must be in UTIL state or higher.

Format

```
>>──ZFRST LOG── ──┬─ALL──────────────────────────────────────────────┬──────────►
                  ├─group────────────────────────────────────────────┤
                  │            ┌─/─────┐                             │
                  ├─MOD── ─────▼─module─┴─────────────────────────────┤
                  │ ┌─/─────────────────────────────────────────┐    │
                  └─▼─mmmmccccchhhhrr── ──THRU── ──mmmmccccchhhhrr─┴────┘

►─ ──damonhhmm── ──THRU── ──damonhhmm─┬─/1──────────┬────────────────────────►◄
                                       └─/processors─┘
```

ALL
: restores the logging records for all the online modules.

group
: is the 1-character letter of the tape group name as defined in the LOGCAP system initialization program (SIP) macro. For example, specify T for RTT or Y for RTY. All records in the range of modules for this tape group name will be restored.

MOD
: restores the logging records for one or more specific online modules.

module
: is a 3-digit hexadecimal symbolic module number.

mmmmccccchhhhrr **THRU** *mmmmccccchhhhrr*
: restores the logging records for a specific address range, where *mmmmccccchhhhrr* is a 14-digit (7-byte) hexadecimal file address:

mmmm
is the 4-digit hexadecimal symbolic module number.

cccc
is the 4-digit hexadecimal cylinder number.

hhhh
is the 4-digit hexadecimal head number.

rr is the 2-digit hexadecimal record number.

damonhhmm **THRU** *damonhhmm*
is the range of time for the logging records that you want to restore, where:

da is a 2-digit number that represents the day of the month.

mon
is the 3-character alphabetic abbreviation for the month.

hhmm
is the time of day in hours and minutes. Use the military time format.

processors
is the number of processors, from 1–8, that were active during record logging.

Additional Information

None.

Examples

Logging records for all the modules are restored in the following example.

```
User:   ZFRST LOG ALL 06MAY0000 THRU 06MAY2359

System: FRST0010I 13.50.26 STARTED
        FRST0029I 13.50.41 LOG RESTORE COMPLETE
        COTC0087A 13.50.41 TCLS HPN    REMOVE XAU FROM DEVICE 422
                           VSN A00242  NOBLK
```

Logging records for the 047 module are restored in the following example.

```
User:   ZFRST LOG MOD 047 06MAY0000 THRU 06MAY2359

System: FRST0057I 13.53.35 MOD 047 ADDED
        FRST0010I 13.53.36 STARTED
        FRST0029I 13.53.51 LOG RESTORE COMPLETE
        COTC0087A 13.53.51 TCLS HPN    REMOVE XAU FROM DEVICE 422
                           VSN A00242  NOBLK
```

Related Information

- See *TPF Database Reference* for more information about restoring record logs.
- See *TPF System Generation* for more information about the LOGCAP macro.

ZFRST PAUSE–Pause the Restore Function

Use this command to temporarily stop (pause) the restore function for the current processor or for all the participating processors. If the restore function is restoring a module (physical DASD unit) when you enter this command, the restore function for that module is completed before the function is paused.

Requirements and Restrictions

Each participating processor must be in UTIL state or higher.

Format

```
►►──ZFRST PAUSE──┬───────┬──────────────────────────────────────────►◄
                 └─ ALL ─┘
```

ALL
temporarily stops the restore function for all the participating processors. If you do not specify the ALL parameter, the restore function is paused only for the current processor.

Additional Information

None.

Examples

The restore function is temporarily stopped for the current processor in the following example.

```
User:   ZFRST PAUSE

System: FRST0043I 12.36.12 PAUSE IN PROGRESS PROCESSOR B
        FRST0009I 12.36.48 MOD 01E ON 0520 FROM TAPE ON 424 - COMPLETED
        FRST0018I 12.36.48 PROCESSOR B PAUSED
        COTC0087A 12.36.48 TCLS WP1    REMOVE BXA FROM DEVICE 424
                           VSN A00245  NOBLK
```

Related Information

See *TPF Database Reference* for more information about the restore function.

ZFRST RESTART–Restart the Restore Function

Use this command to start the restore function again after you temporarily stop it using the ZFRST PAUSE command. You can also use this command to restart the restore function after an initial program load (IPL) is performed for the TPF system.

Requirements and Restrictions

- Load the restore tapes (BXA–BXZ, BX0–BX5) and ensure that they are ready to use.
- Each participating processor must be in UTIL state or higher.

Format

```
►►──ZFRST RESTART──┬───────┬──────────────────────────────────────────►◄
                   └─ ALL ─┘
```

ALL
starts the restore function again for all the participating processors. If you do not specify the ALL parameter, the restore function starts again only for the current processor.

Additional Information

None.

Examples

The restore function is started again for the current processor in the following example.

```
User:   ZFRST RESTART

System: FRST0014I 11.16.30 PROCESSOR B RESTART COMPLETE
```

Related Information

See *TPF Database Reference* for more information about the restore function.

ZFRST STATUS–Display Restore Status

Use this command to display the status and progress of the restore function.

Requirements and Restrictions

None.

Format

```
►►──ZFRST STATUS──┬──────────────────────────────────┬──────────────────►◄
                  └──MODULE──  ──┬─module─────┬──┘
                                 ├─ALL────────┤
                                 ├─COMPLETED──┤
                                 ├─INPROGRESS─┤
                                 └─REMAINING──┘
```

MODULE
displays information that indicates whether the restore function completed, is in progress, or was not started yet for a specific module (physical DASD unit). If you do not specify the MODULE parameter, overall status information about the restore function is displayed.

module
is a 3-digit hexadecimal symbolic module number. When specified, information is displayed that indicates whether the restore function completed, is in progress, or was not started yet for that module.

ALL
lists all the modules and indicates, for each module, whether the restore function completed, is in progress, or was not started yet.

COMPLETED
lists the modules where the restore function completed.

INPROGRESS
lists the modules where the restore function is in progress.

REMAINING
lists the modules where the restore function was not started yet.

Additional Information

None.

Examples

Overall status information about the restore function is displayed in the following example.

```
User:   ZFRST STATUS

System: FRST0001I 15.37.59
        RESTORE STARTED 06MAY 1536
        MODULES TO DO        2
        MODULES COMPLETED    0
        MODULES REMAINING    2
                               PROCESSOR
                               B
        MAXIMUM ECB ALLOWED   032
        CURRENT MAXIMUM ECB   016
        CURRENT ACTIVE ECB    002
        XCP RECORDING ACTIVE
        DISK ERRORS
        TRKS 000
        RCDS 000
        TAPE ERRORS 000
            TAPE DEVICE    - STATUS -    PROCESSOR
                             KPT  /  CP
          BXA   423   424   80   / 00 03     B
          BXB   480   481   C0   / 00 03     B
        INPROGRESS
        MOD 01E ON DEV 0520 FROM TAPE 423 STARTED 1536
        MOD 01F ON DEV 0521 FROM TAPE 480 STARTED 1536
        END OF STATUS DISPLAY
```

Status information about the restore function for module 01E is displayed in the following example.

```
User:   ZFRST STATUS MODULE 01E

System: FRST0126I 15.39.22  MOD 01E IS INPROGRESS
```

Status information about all the modules is displayed in the following example.

```
User:   ZFRST STATUS MODULE ALL

System: FRST0002I 15.40.10 RESTART MODULE STATUS
        FRST - MODULES COMPLETED
                NO MODULES FOUND
        FRST - MODULES INPROGRESS
            01E 01F
        FRST - MODULES REMAINING
                NO MODULES FOUND
        DISPLAY COMPLETE
```

Related Information

See *TPF Database Reference* for more information about the restore function.

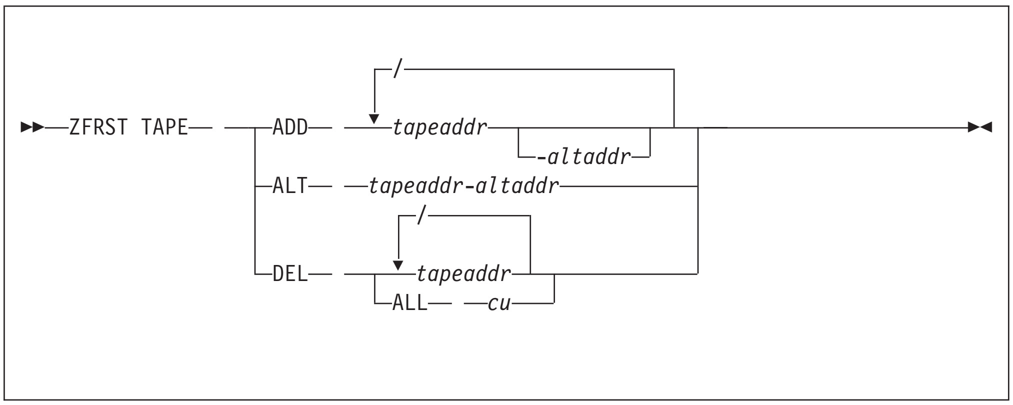
ZFRST TAPE–Add or Delete Tape Devices

Use this command to make additional primary and alternate tape devices available to the restore function or to delete primary tape devices (and the corresponding alternate tape devices, if they exist). After a tape device is deleted, it is no longer available to the restore function.

Requirements and Restrictions

- Enter this command from the processor where you want to add or delete the tape device.
- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format



ADD
adds primary tape devices and corresponding alternate tape devices.

ALT
adds alternate tape devices for existing primary tape devices.

DEL
deletes primary tape devices and the corresponding alternate tape devices, if they exist.

tapeaddr
is the 3-digit hexadecimal address of a primary tape device.

altaddr
is the 3-character hexadecimal address of an alternate tape device.

Note: The addresses of the primary and alternate tape devices must be on the same device type.

ALL *cu*
deletes all of the primary and alternate tape devices on the specified control unit, where *cu* is the 2-digit hexadecimal address of the control unit.

Additional Information

If you delete a tape device while it is being used to restore a module, load the tape in another tape device to continue the restore function.

Examples

A primary tape device and an alternate tape device are added for the restore function in the following example.

```
User:   ZFRST TAPE ADD 480-481

System: FRST0119I 14.50.19 DUAL TAPE 480-481 ADDED
```

An alternate tape device is added for tape devices 423 and 480 in the following example.

```
User:   ZFRST TAPE ALT 423-424/480-481

System: FRST0119I 15.34.37 DUAL TAPE 423-424 ADDED
        FRST0119I 15.34.37 DUAL TAPE 480-481 ADDED
```

Primary tape device 480 and its corresponding alternate tape device (if it exists) are deleted in the following example. These tape devices are no longer available to the restore function.

```
User:   ZFRST TAPE DEL 480

System: FRST0120I 14.52.01 DUAL TAPE 480-481 DELETED
        FRST0016I 14.52.01 MOD 01F ON 0521 FROM TAPE ON 480 - ABORTED
        COTC0087A 14.52.01 TCLS WP1    REMOVE BXB FROM DEVICE 480
                           VSN A00248  NOBLK
```

Related Information

See *TPF Database Reference* for more information about the restore function.

ZFRST XCP–Restore Exception Records

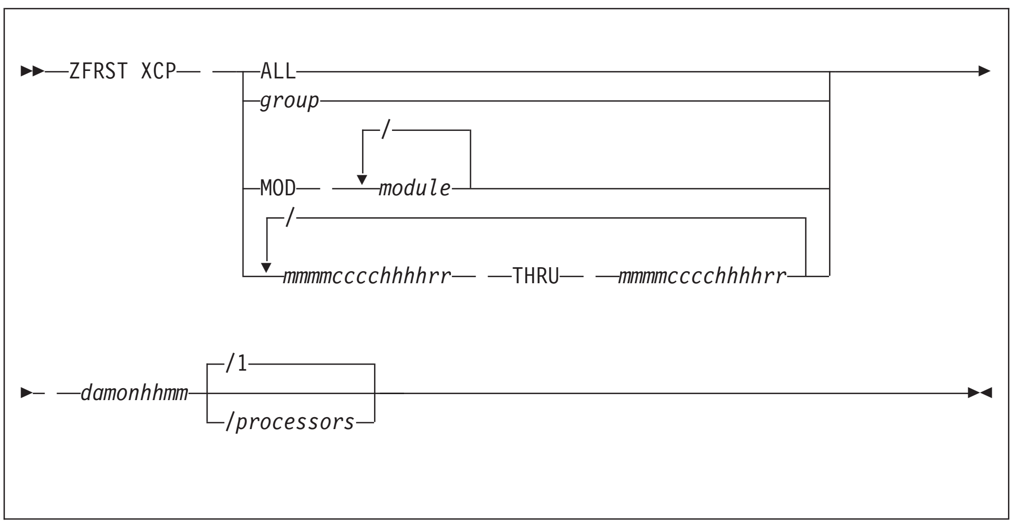
Use this command to restore exception records that were captured starting at a specific time. You can restore exception records for all the online modules (physical DASD units), for specific online modules, or for a specific address range.

Requirements and Restrictions

- Use the ZTMNT command to mount the exception tapes (XAT–XAY) for input for each processor. Specify the BP parameter.

  If more than one processor was active when the exception records were captured, you must mount the logging tapes as follows:
  - Mount the exception recording tapes (RTT–RTY) for processor 1 as XAT–XAY, respectively.
  - Mount the exception recording tapes (RTT–RTY) for processor 2 as XBT–XBY, respectively.
  - Mount the exception recording tapes (RTT–RTY) for processor 3 as XCT–XCY, respectively, and so on.
- Each participating processor must be in UTIL state or higher.
- Do not use the ALL prefix with this command.

Format



ALL
restores the exception records for all of the online modules.

group
is the 1-character letter of the tape group name as defined in the LOGCAP system initialization program (SIP) macro. For example, specify T for RTT or Y for RTY. All records in the range of modules for this tape group name will be restored.

MOD
restores the exception records for one or more specific online modules.

module
is a 3-digit hexadecimal symbolic module number.

mmmmccccchhhhrr **THRU** *mmmmccccchhhhrr*
restores the exception records for a specific address range, where *mmmmccccchhhhrr* is a 14-digit (7-byte) hexadecimal file address:

mmmm
is the 4-digit hexadecimal symbolic module number.

cccc
is the 4-digit hexadecimal cylinder number.

hhhh
is the 4-digit hexadecimal head number.

rr is the 2-digit hexadecimal record number.

damonhhmm
is the starting time stamp. All exception records captured after the specified time stamp are restored, where:

da is a 2-digit number that represents the day of the month.

mon
is the 3-character alphabetic abbreviation for the month.

hhmm
is the time of day in hours and minutes. Use the military time format.

processors
is the number of processors, from 1–8, that were active during exception recording.

Additional Information

None.

Examples

Exception recording records for all of the modules are restored in the following example.

```
User:   ZFRST XCP ALL 06MAY0000

System: FRST0010I 12.28.03 STARTED
        FRST0029I 12.28.04 XCP RESTORE COMPLETE
        COTC0087A 12.28.04 TCLS WP1    REMOVE XAU FROM DEVICE 422
                           VSN A00242  NOBLK
```

Exception recording records for modules 01E and 01F are restored in the following example.

```
User:   ZFRST XCP MOD 01E/01F 06MAY0000

System: FRST0057I 12.29.44 MOD 01E ADDED
        FRST0057I 12.29.44 MOD 01F ADDED
        FRST0010I 12.29.45 STARTED
        FRST0029I 12.29.45 XCP RESTORE COMPLETE
        COTC0087A 12.29.45 TCLS WP1    REMOVE XAU FROM DEVICE 422
                           VSN A00242  NOBLK
```

Related Information

- See *TPF Database Reference* for more information about exception recording.
- See *TPF System Generation* for more information about the LOGCAP macro.

ZGAFA–Get File Pool Address

Use this command to:

1. Get a file address from a long-term pool.
2. Set the record at the file address to 0.
3. Add a record ID to the header of the record. You can use this record ID to perform subsequent file operations on the record at the assigned file address.

Requirements and Restrictions

None.

Format

```
►►──ZGAFA── ──device── ──pool── ──recid──────────────────────────────────►◄
```

device
: is a 4-character alphanumeric device name as defined in the ONLFIL SIP macro (for example, 3380 or DEVA).

pool
: is one of the following pool types:

SLT
Small, long-term pool

LLT
Large, long-term pool

SDP
Small, long-term duplicate record pool

LDP
Large, long-term duplicate record pool.

Note: The duplicate record pools are valid only for partially duplicated databases.

recid
: is a 2-character alphanumeric record ID or a 4-digit hexadecimal record ID.

Additional Information

This command can be used only to obtain large or small, long-term pool addresses. Use the ZGAFI command to obtain 4-KB file pool addresses.

Examples

In the following example, a file address is obtained from the large, long-term duplicate record pool.

```
User:   ZGAFA DEVA LDP 00AB

System: GAFA0001I 09.05.41 GAFA DEVA LDP F.A.-000C0001
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGAFI–Get File Pool Address by ID

Use this command to retrieve a record ID attribute table (RIAT) record. This function calls the GETFC macro to obtain a file address from the specified pool, sets the record to zero, and files the record to the designated address.

Requirements and Restrictions

Do not enter this command while the TPF system is in 1052 or UTIL state because the get file storage (GFS) facility is not active in these states.

Format

```
                         ┌─ ─0─────┐
►►──ZGAFI── ──recid──────┼─────────┼─────────────────────────────────────►◄
                         ├─ ─riat──┤
                         ├─ ─P─────┤
                         └─ ─O─────┘
```

recid
is a 2-character alphanumeric record ID or a 4-digit hexadecimal record ID.

riat
is the RIAT type from 0–9.

P corresponds to RIAT type 0.

O corresponds to RIAT type 1.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZGAFI HELP

  ZGAFI ?
- In previous releases of the TPF system, the P parameter specified the prime RIAT record and the O parameter specified the overflow RIAT record. These records are now referred to as type 0 and type 1 records, respectively. However, the P and O parameters are still provided for migration purposes.
- If you do not specify the *riat* parameter variable, or the P or O parameter, the default RIAT type is 0.

Examples

In the following example, a character record ID is specified to retrieve a RIAT record.

```
User:   ZGAFI OM 1

System: GAFI0008I 09.06.50 GAFI C OM 1 FILE ADDRESS - 00000000001004BC
```

In the following example, a hexadecimal record ID is specified to retrieve a RIAT record.

```
User:   ZGAFI 00F3 0

System: GAFI0008I 09.07.31 X 00F3 0 FILE ADDRESS-00000000001819C9
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP–Alter Allowed Use

Use this command to change the allowed use-per-minute for long-term file pools.

Requirements and Restrictions

None.

Format

```
                          (1)
►►──ZGFSP── ──┬─SLU─┬───────── ──device── ──value──────────────────────────►◄
              ├─SDU─┤
              ├─LLU─┤
              ├─LDU─┤
              ├─4LU─┤
              ├─4DU─┤
              └─46U─┘
```

Notes:

1 This parameter is not valid until a **ZPMIG CONVERT** request is processed successfully. However, after the pool format conversion is complete, you ***must*** specify this parameter.

SLU
changes the allowed use for small, long-term (SLT) file pools.

SDU
changes the allowed use for small, long-term duplicate (SDP) file pools.

LLU
changes the allowed use for large, long-term (LLT) file pools.

LDU
changes the allowed use for large, long-term duplicate (LDP) file pools.

4LU
changes the allowed use for 4-KB long-term (4LT) file pools.

4DU
changes the allowed use for 4-KB long-term duplicate (4DP) file pools.

46U
changes the allowed use for 4-KB long-term duplicate FARF6 (4D6) file pools.

device
is the name of the device type associated with the pool section that will be monitored.

value
is a 9-digit decimal value that represents the number of records per minute.

Additional Information

None.

Examples

The allowed file use for small, long-term file pools is changed in the following example.

```
User:   ZGFSP SLU DEVA 000020

System: GFSP0005I 09.14.12 SLU DEVA 000020-PERFORMED
```

Related Information

- See *TPF Database Reference* for more information about file pool support.
- See “ZPMIG–Migrate Pool Structures” on page 1116 for more information about converting pool data structures.

ZGFSP–Alter Minimum Count

Use this command to change the minimum number of long-term file pools that can be available before an attention message is issued.

Requirements and Restrictions

None.

Format

```
                          (1)
►►──ZGFSP── ──┬─SLM─┬──────── ──device── ──value────────────────────────────►◄
              ├─SDM─┤
              ├─LLM─┤
              ├─LDM─┤
              ├─4LM─┤
              ├─4DM─┤
              └─46M─┘
```

Notes:

1 This parameter is not valid until a **ZPMIG CONVERT** request is processed successfully. However, after the pool format conversion is complete, you ***must*** specify this parameter.

SLM
changes the minimum count for small, long-term (SLT) file pools.

SDM
changes the minimum count for small, long-term duplicate (SDP) file pools.

LLM
changes the minimum count for large, long-term (LLT) file pools.

LDM
changes the minimum count for large, long-term duplicate (LDP) file pools.

4LM
changes the minimum count for 4-KB long-term (4LT) file pools.

4DM
changes the minimum count for 4-KB long-term duplicate (4DP) file pools.

46M
changes the minimum count for 4-KB long-term duplicate FARF6 (4D6) file pools.

device
is the name of the device type associated with the pool section that will be monitored.

value
is a 9-digit decimal value that represents the minimum count.

Additional Information

None.

Examples

The minimum count for 4-KB, long-term duplicate file pools is changed in the following example.

```
User:   ZGFSP 4DM DEVA 000010

System: GFSP0005I 09.14.52 4DM DEVA 000010-PERFORMED
```

Related Information

- See *TPF Database Reference* for more information about file pool support.
- See “ZPMIG–Migrate Pool Structures” on page 1116 for more information about converting pool data structures.

ZGFSP–Alter Pool Parameters

Use this command to change the parameters for file pool records.

Requirements and Restrictions

None.

Format

```
►►──ZGFSP── ──┬─REJ─┬── ──value──────────────────────────────────────►◄
              ├─KUL─┤
              ├─4RT─┤
              ├─LRT─┤
              ├─SRT─┤
              ├─SKP─┤
              └─STR─┘
```

REJ
defines the rejection value.

Reject is used for long-term reorder. Specify the value as a percentage; it is used to calculate the minimum number of addresses that must be available for a set to be accepted by reorder and put to use.

As a reorder for a long-term section takes place, the total number of addresses currently in the section is divided by the number of directories to generate the average number of addresses per directory. The number of addresses in the set just brought in by reorder must exceed the:

```
Average number of addresses per directory
                ×
The number of directories in this set
                ×
          The reject percentage
```

KUL
defines the keypoint update level. Keypointing a directory occurs when the specified number of changes is made to the directory by dispense or return pool activity.

4RT
defines the 4-KB, short-term pool section recycle time increment in minutes.

LRT
defines the large, short-term pool section recycle time increment in minutes.

SRT
defines the small, short-term pool section recycle time increment in minutes.

SKP
defines the file pool address skipping safety factor. This value should reflect the number of addresses that could have been dispensed between a keypoint request and the actual update of the directory on file.

When a processor restarts from an unplanned shutdown, the keypoint images of the in-use directories do not reflect all the addresses that were dispensed before the unplanned shutdown. To ensure that an address that was previously dispensed is not given out again, the TPF system adds the values assigned to the KUL and SKP parameters and discards this count from each of its directories.

STR
defines the short-term directory reorder level. This is a percentage of the number of addresses that exist when all directories in a set are full. When the count of available addresses in the set falls below this value, a reorder for a new set is started.

value
is the decimal time or file pool address factor, which is a 1- to 2-digit value for the REJ and STR parameters, and a 1- to 4-digit value for the other parameters.

Additional Information

The *short-term pool section recycle time* values are the number of minutes a recently depleted short-term directory should remain unused to allow any dispensed addresses from this directory to be given up by applications. There is one recycle time value per short-term pool type.

Examples

The keypoint update level is changed in the following example.

```
User:   ZGFSP KUL 0040

System: GFSP0005I 09.15.11 KUL 0040-PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP DSP–Display General File Storage Parameters

Use this command to display the following get file storage (GFS) information:
- Value of the pool record parameters
- Allowed use per minute for long-term file pool records
- Minimum count of long-term file pool records
- Fallback schedules
- Ratio dispensing schedules
- Size of the pool record directory sets.

Requirements and Restrictions

None.

Format

```
►►──ZGFSP DSP──┬───────┬──────────────────────────────────────────────►◄
               └─ ─SET─┘
```

SET
displays the size of the pool directory sets. If you do not specify this parameter, the other GFS information is displayed.

Additional Information

None.

Examples

Information about the get file storage (GFS) facility is displayed in the examples that follow. The following example shows the display ***before*** a successful pool format conversion (ZPMIG CONVERT request).

```
User:   ZGFSP DSP

System: GFSP0002I 09.15.30 GFS PARAMETER DISPLAY

        KUL  SKP  4RT  LRT  SRT  STR  REJ  OPTIONS
        0040 0040 0150 0150 0250 0010 0010 NOMON NOLOG NOTAG

        USLT       USDP       ULLT       ULDP       U4LT       U4DP
        0000000020 0000000000 0000000000 0000000000 0000000000 0000000010

        MSLT       MSDP       MLLT       MLDP       M4LT       M4DP
        0000000000 0000000000 0000000000 0000000000 0000000000 0000000000

        FALLBACK SCHEDULES
        SST DEVA
        SDP DEVA
        LST DEVA
        LDP DEVA
        4ST DEVA
        4DP DEVA

        RATIO DISPENSING SCHEDULES
        SST DEVA-99
        SDP DEVA-99
        LST DEVA-99
        LDP DEVA-99
        4ST DEVA-99
        4DP DEVA-99

        END ZGFSP DISPLAY
```

The following example shows the display ***after*** a successful pool format conversion (ZPMIG CONVERT request).

```
User:   ZGFSP DSP

System: GFSP0006I 09.15.30 GFS PARAMETER DISPLAY

        KUL  SKP  4RT  LRT  SRT  STR  REJ  OPTIONS
        0040 0040 0150 0150 0250 0010 0010 NOMON NOLOG NOTAG

                  MAX USAGE PER MINUTE
        USLT DEVA                   20
        USDP DEVA                    0
        ULLT DEVA                    0
        ULDP DEVA                    0
        U4LT DEVA                    0
        U4DP DEVA                   10
        U4D6 DEVA                    0

                  MIN AVAILABLE
        MSLT DEVA                    0
        MSDP DEVA                    0
        MLLT DEVA                    0
        MLDP DEVA                    0
        M4LT DEVA                    0
        M4DP DEVA                    0
        M4D6 DEVA           5000000000

        FALLBACK SCHEDULES
        SST DEVA
        SDP DEVA
        LST DEVA
        LDP DEVA
        4ST DEVA
        4DP DEVA
        4D6 DEVA

        RATIO DISPENSING SCHEDULES
        SST DEVA-99
        SDP DEVA-99
        LST DEVA-99
        LDP DEVA-99
        4ST DEVA-99
        4DP DEVA-99
        4D6 DEVA-99

        END ZGFSP DISPLAY
```

Information about the size of the pool directory sets is displayed in the following example.

```
User:   ZGFSP DSP SET

System: GFSP0003I 09.16.27 GFS SET SIZE DISPLAY

                        ACTIVE  STANDBY NEW     CARVE

             SST  DEVA  0001    0001    0001    0001
             SDP  DEVA  0001    0001    0001    0001
             LST  DEVA  0001    0001    0001    0001
             LDP  DEVA  0001    0001    0001    0001
             4ST  DEVA  0001    0001    0001    0001
             4DP  DEVA  0001    0001    0001    0001
             4D6  DEVA  0001    0001    0001    0001

        END ZGFSP DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP FLB–Specify a Pool Fallback Schedule

Use this command to specify a pool fallback schedule.

Requirements and Restrictions

None.

Format

```
                                 ┌─/──────────────┐
                                 │  (1)           │
►►──ZGFSP FLB── ──pool── ────────▼──────────device─┴─┬──────────────────────────►◄
                              └─CLEAR────────────────┘
```

Notes:

1 You must specify from 2–4 devices.

pool
is one of the following pool type codes:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

device
is the name of a device.

CLEAR
clears the schedule for the specified pool type.

Additional Information

None.

Examples

A pool fallback schedule is specified for small, short-term pools in the following example.

```
User:   ZGFSP FLB SST 3350/3380

System: GFSP0005I 09.18.16 FLB SST 3350/3380-PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP OPT–Set Get File Storage (GFS) Options

Use this command to set the get file storage (GFS) options.

Requirements and Restrictions

None.

Format

```
►►──ZGFSP OPT── ──┬────┬──┬─LOG─┬────────────────────────────────────►◄
                  └─NO─┘  ├─TAG─┤
                          └─MON─┘
```

NO
turns off the following options.

LOG
starts logging to tape all dispensed file pool addresses with the name of the program that is requesting each address.

TAG
starts tagging all long-term released file pool addresses with the name of the releasing program.

MON
starts monitoring long-term file pools on each processor that is in NORM state in a loosely coupled environment. If a processor is not in NORM state when you enter this command, the pool monitor automatically starts on that processor when the processor is cycled to NORM state. The pool monitor is unique per processor.

Additional Information

None.

Examples

The MON option is turned on in the following example. While the monitor was running, more than the established maximum number of pools were used.

```
User:   ZGFSP OPT MON

System: GFSP0005I 14.17.54 OPT MON -PERFORMED

        CYGM0011W 14.23.00 SDP POOL USE HAS EXCEEDED LIMIT. 374 ADDRESSES USED
        IN LAST MIN.
```

The MON option is turned off in the following example.

```
User:   ZGFSP OPT NOMON

System: GFSP0005I 09.19.17 OPT NOMON -PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP RCY–Recycle Short-Term Pool Directory Record

Use this command to allow short-term pool directory records to be recycled.

Requirements and Restrictions

You must enter this command from the processor where you want to recycle the short-term pool directory records.

Format

```
►►──ZGFSP RCY── ──device── ──pool──────────────────────────────────────►◄
```

device
is the name of the device.

pool
is one of the following pool type codes:

SST
Small, short-term pool

LST
Large, short-term pool

4ST
4-KB, short-term pool.

Additional Information

None.

Examples

Large, short-term pool directory records can be recycled in the following example.

```
User:   ZGFSP RCY 3350 LST

System: ZGFSP RCY DEVA LST
        CYC60002I 13.13.35 POOL TYPE LST DEVICE DEVA COUNTS      200 IN USE
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP RTO–Specify a Ratio Dispensing Schedule

Use this command to specify a ratio dispensing schedule.

Requirements and Restrictions

The ratio dispensing schedule does not take effect until the get file storage (GFS) facility is stopped and started again. You can stop and restart GFS by cycling the TPF system to 1052 state and then back to NORM state.

Format

```
                              ┌─/──────────────────┐
                              │  (1)               │
►►──ZGFSP RTO── ──pool── ──┬──▼──────device-number─┴─┬──────────────────────►◄
                           └─CLEAR───────────────────┘
```

Notes:

1 You must specify from 2–4 devices.

pool
is one of the following pool type codes:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

device
is the name of a device as defined in the ONLFIL SIP macro (for example, 3380 or DEVA).

number
is the number of file pool addresses to be dispensed from the specified device before selecting another pool section.

CLEAR
clears the schedule for the specified pool type.

Additional Information

None.

Examples

A ratio dispensing schedule is specified for small, long-term pool records in the following example.

```
User:   ZGFSP RTO SLT 3350-50/3380-50

System: GFSP0005I 09.21.35 RTO SST 3350-50/3380-50-PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZGFSP SET–Change Size of Pool Directory Set

Use this command to change the size of a pool directory set.

Requirements and Restrictions

None.

Format

```
►►─ZGFSP SET── ─pool── ─size─┬──────────────────────┬──────────────►◄
                             └─ ─cpuid─┬───────────┬┘
                                       └─ ─device─┘
```

pool
is one of the following pool types codes:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

size
is the number of directories in the set from 1–255.

cpuid
is the 1-character alphanumeric CPU ID of a processor.

device
is the name of a device.

Additional Information

If you do not specify a value for the *cpuid* and *device* variables, the pool directory set size for all active devices and processors is changed.

Examples

The size of small, long-term pool directory sets is changed in the following example.

```
User:   ZGFSP SET SLT 5 C 3380

System: GFSP0005I 09.22.20 SET SLT 5 C 3380-PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZIDOT–Display or Modify Dump Override Table

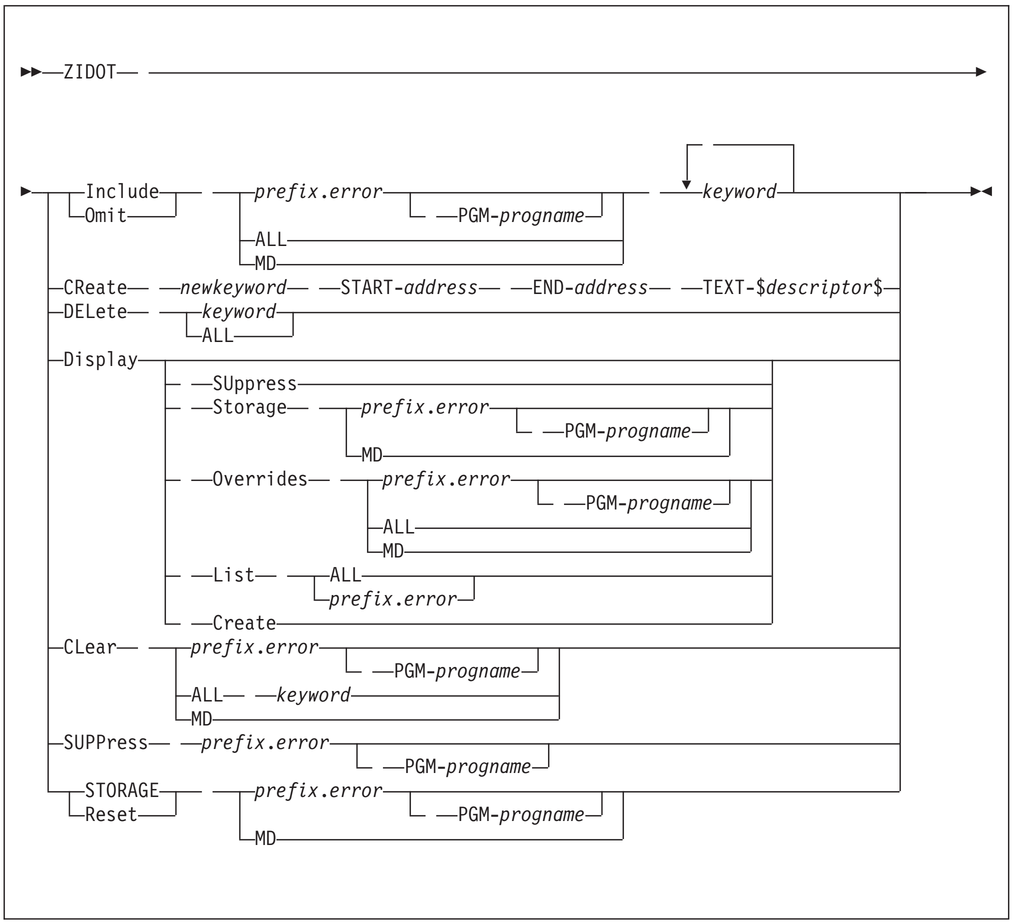
Use this command to:

- Specify areas of main storage to be included in or omitted from the dump for a particular system error
- Define a new storage area that is normally omitted from system storage dumps and associate a keyword with the storage area
- Bypass the processing of a particular system error or force a dump of the entire system virtual memory (SVM)
- Display the current storage or overrides for a system error
- Display the errors being suppressed or overridden.

Requirements and Restrictions

None.

Format



Include
: includes the specified areas of main storage in the dump for the specified system error.

If you specify a keyword that was created using this command, the storage area associated with the keyword is printed in both hexadecimal and EBCDIC format in the dump.

Omit
omits the specified areas of main storage from the dump for the specified system error.

prefix.error
is the prefix and 6-digit system error number as it displays in *Messages (System Error and Offline)* and *Messages (Online)*. Leading zeros can be omitted.

Note: Specify an asterisk (*) for the prefix if no prefix is associated with error number. This condition occurs when the dump is issued from the `perror` or `exit` C function.

ALL
overrides the list of storage areas included in all dumps. Overrides entered with this parameter take precedence over any conflicting overrides entered for a specific system error number.

MD
overrides the list of storage areas included in a selective manual dump.

PGM-*progname*
is the name of an ECB-controlled program. The override takes effect only if the system error occurred in the specified program.

keyword
is one or more keywords that identify a particular storage area. The keywords are defined in the IDATB macro. For more information about the IDATB macro, see *TPF System Macros*.

You can use the ZIDOT DISPLAY command to display a list of valid keywords.

CReate
creates a new keyword and associates a storage range and printable description with it.

Note: The storage associated with the keyword is not included in dumps until you enter the ZIDOT command and specify the INCLUDE parameter.

newkeyword
is a new keyword that does not exceed 36 alphanumeric characters.

START-*address* **END-***address*
are starting and ending addresses for a range of storage. The addresses must be a valid range in the system virtual memory (SVM) of the main I-stream. Addresses from X'00'–X'FFF' are not accepted.

TEXT=*$descriptor$*
is a description for the new keyword. You can substitute a single quotation (') for the dollar sign ($).

DELete *keyword*
deletes a keyword you created using this command.

DELete ALL
deletes all the keywords that you created using this command.

Display
displays a list of all valid keywords, including those created using this command.

The addresses of the storage associated with each keyword are also displayed. If a keyword is subsystem-unique, the address for the basic subsystem (BSS) is displayed. If the keyword is I-stream unique, the address for the main I-stream is displayed.

Display SUppress
displays a list of all the system errors that were suppressed.

Display Storage
displays the list of storage areas that are included in the dump when the specified system error occurs.

Note: If you specify the PGM parameter, the storage that is displayed is the storage for the program without any specific version code.

Display Overrides
displays the storage areas that were included or omitted from the specified dump using this command.

Display List *prefix.error*
displays a list of all system error number and program name combinations for the specified system error for which overrides exist.

Display List ALL
displays a list of system error numbers and program name combinations for which overrides exist.

Display Create
displays a list of all the keywords that were created using the ZIDOT command.

CLear
clears the dump override table entry for the specified system error. Any overrides created using this command are removed.

CLear ALL *keyword*
cancels an INCLUDE ALL or OMIT ALL request. The specified storage is no longer included in or omitted from all dumps.

SUPPress
suppresses the dump for the specified system error.

STORAGE
forces a dump of the entire system virtual memory (SVM) for the specified system error. This takes place for only one dump.

Note: In a production environment, a full dump could cause the TPF system to be masked for several minutes and cause a large spike in system activity when the dump ends.

Reset
resets the SUPPRESS or STORAGE parameter for the specified system error. If a list of keywords was previously in effect, the list is restored. Otherwise the dump override table entry is cleared.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIDOT HELP
  ZIDOT ?

- An entry in the dump override table associates a system error with a list of one or more storage areas to be included in the dump for the system error. The storage areas are identified by user-defined keywords.
- The facility for including or omitting storage areas from individual dumps is intended for use as a temporary bypass. Permanent dump overrides should be installed in the TPF system by coding one or more IDOTB macro calls in CUDP copy member of the CCUEXT user exit (CIDP in CCCPSE for IBM overrides).
- Entries created using this command are processor- and image-unique. Performing an IPL with a different image will cause entries created on the current image to be ignored. See the IDOTB macro description in *TPF System Macros* for more information.
- Dump overrides have no effect on OPR dumps unless you set the LONG option for the ZASER command, or unless a particular SERRC macro has the SYSDUMP=YES parameter coded. Conversely, system storage is always included in CTL dumps except when SYSDUMP=NO is coded for the SERRC macro.
- Overrides for full manual dumps (OPR-I030000) are ignored by system error processing. Overrides for specially defined system errors that never produce a dump (for example, ZRIPL) are also ignored.
- If a static override was defined for the system error using the IDOTB macro, this command uses this override as a starting point. Storage areas included using the IDOTB macro can be selectively omitted using this command. Overrides created using this command take priority over those created using the IDOTB macro.
- Some storage areas may have been designated for inclusion in every dump when they were defined (REQUIRED=YES parameter on the IDATG macro). These areas are included by default whenever a dump override is created. The inclusion of these default areas cannot be overridden using this command.
- You can create overlapping overrides using this command or other overrides in the TPF system. When overlapping overrides are created using this command, overlapping areas will only be dumped once.
- If a specified storage area has been defined as MDBF subsystem-unique or I-stream unique, all copies of the data structure are included in the dump. However, if a static override exists and the storage area is included in the static override, all copies are included only if COPIES=ALL was coded for the IDOTB macro that was used to define the static override.
- Selective memory dump table (SMDT) entries generated by IDATG KEYWORD=DUMMY macro calls are used to store the information for created keywords. The number of created keywords that are allowed is determined by the number of DUMMY entries allocated using IDATG.
- The ZIDOT command uses bit maps to identify storage areas to be included or omitted. These bit maps depend on the arrangement of the IDATG macro calls that are used to define the dump keywords (see the IDATG macro description in *TPF System Macros*). If the number or ordering of the defined keywords changes, the ZIDOT bit maps will no longer be valid.
- As a precaution, restart program CVRN compares the number of defined keywords with the number defined before the last IPL of the TPF system. If the number of keywords has changed, all ZIDOT overrides are deleted and the ZIDOT records on file are reinitialized.
- Changing the order of the keywords after the ZIDOT function is used is not recommended. However, if it cannot be avoided, the ZAREC command can be used to clear the record ID from the dump override table file records (see the SYSEQ macro). This will clear and reinitialize the records during the next IPL of the TPF system.

Examples

In the following example, the control program and the CLH control tables are included in system error I00000D.

```
User:   ZIDOT INCLUDE I.00000D ICP ICLH

System: IDOT0002I 14.26.27 KEYWORD ICP INCLUDED FOR ERROR I00000D
        IDOT0002I 14.26.27 KEYWORD ICLH INCLUDED FOR ERROR I00000D
        IDOT0001I 14.26.27 COMPLETED
```

In the following example, the ABCD keyword is created for the storage area beginning at address X'1F0000' and ending at address X'1F0FFF'. This storage area is included in all dumps.

```
User:   ZIDOT CR ABCD START-1F0000 END-1F0FFF TEXT-$ALLOCATION TABLE SUBSET$

System: IDOT0004I 14.26.27 KEYWORD ABCD CREATED START-001F0000 END-001F0FFF
```

The following example displays a list of the storage areas that were included or omitted from the dump for the I00000D system error using this command.

```
User:   ZIDOT D OVERRIDES I.D

System: IDOT0013I 13.39.46 OVERRIDES VIA ZIDOT FOR ERROR I00000D
        ICP      - CONTROL PROGRAM                    *INCLUDED*
        ICLH     - CLH BLOCK MGMT TABLES              *INCLUDED*
        IDOT0011I 13.39.46 DISPLAY COMPLETE
```

Related Information

- See *TPF Main Supervisor Reference* for more information about the system error dumps.
- See also the SERRC macro and the SNAPC macro in *TPF General Macros* for more information.

ZIFIL–Initialize Fixed File Data Records

Use this command to initialize fixed file data records.

The first record (starting ordinal number for initialization) can be preformatted and used to propagate a data organization throughout the remaining records that are being initialized.

Requirements and Restrictions

None.

Format

```
►►──ZIFIL──equate──/──recid──/──code──/──start──/──end──/──┬─NNN─┬──────────►
                                                           ├─AFN─┤
                                                           ├─AFB─┤
                                                           ├─ANB─┤
                                                           ├─OFN─┤
                                                           ├─OFB─┤
                                                           └─ONB─┘

►─/──┬─N─┬──┬──────────────────────────────────┬──────────────────────────►◄
     └─C─┘  └─/──istream──/──cpuid──/──ssuser─┘
```

equate
is a 1- to 8-character SYSEQ label (for FACS).

Note: The number sign (#) that precedes the SYSEQ label is optional unless you need to distinguish between 2 labels; for example, #RECA and RECA.

recid
is a 2-character alphanumeric record ID or a 4-digit hexadecimal record ID.

code
is the 2-digit hexadecimal value for the record code check byte for the records.

start
is the 1- to 8-digit decimal starting ordinal number for the initialization.

end
is the 1- to 8-digit decimal ending ordinal number for the initialization.

NNN
specifies the no chaining option for initialization.

AFN
specifies the address value forward chaining option for initialization.

AFB
specifies the address value forward and backward chaining option for initialization.

ANB
specifies the address value backward chaining option for initialization.

OFN
specifies the ordinal number forward chaining option for initialization.

OFB
specifies the ordinal number forward and backward chaining option for initialization.

ONB
specifies the ordinal number backward chaining option for initialization.

N specifies that no copy record is provided. When you specify this parameter, the record ID of the retrieved record is overlaid with the ID specified in this command.

C specifies that this is the first record data (starting ordinal number) to be copied to all initialized records.

Note: When you specify this parameter, the record ID must match the ID in the retrieved record.

istream
is a decimal number, from 1 to 16, of the specified I-stream.

cpuid
is the 1-character alphanumeric CPU ID of a processor.

ssuser
is a 1- to 4-character alphanumeric subsystem user name.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIFIL HELP
  ZIFIL ?
- This function issues a DLAYC macro after filing each record, which can have an impact on performance.

Examples

PDREC fixed file records, that are not preceded by a number sign (#), are initialized in the following example.

```
User:   ZIFIL PDREC/00E1/00/0/18/NNN/N

System: IFIL0004I 16.09.57 RECORD INITIALIZATION STARTED FOR PDREC
        IFIL0005I 16.09.57 RECORD INITIALIZATION COMPLETED FOR PDREC
```

#PDREC fixed file records are initialized in the following example.

Notes:

1. The number sign (#) character has been substituted with a period (.) because some terminals do not support the number sign.
2. In this example, if the PDREC record type does not exist, then entering a ZIFIL command with a PDREC or #PDREC record type will initialize the #PDREC fixed file records.

```
User:   ZIFIL #PDREC/00E1/00/0/18/NNN/N

System: IFIL0004I 16.09.57 RECORD INITIALIZATION STARTED FOR .PDREC
        IFIL0005I 16.09.57 RECORD INITIALIZATION COMPLETED FOR .PDREC
```

Related Information

See *TPF Main Supervisor Reference* for more information about initializing file records.

ZIMAG CLEAR–Clear a TPF Image

Use this command to clear an image or initial program load (IPL) definition from the system. This message *undefines* the image or IPL area and marks it as empty.

Requirements and Restrictions

- You cannot clear an IPL area if it is referenced by an enabled image.
- You must disable an image before you can clear it. If the image has logical references from other images, you must also disable those other images.
- You must use the BP parameter when clearing an image if there are logical references to that image from other images. If you do not specify the BP parameter when it is required, a message is displayed that indicates the logical references exist.

Format

```
►►──ZIMAG CLEAR── ──┬─image──┬──────┬─┬──────────────────────────────────►◄
                    │        └─ ──BP─┘ │
                    └─IPL── ──i────────┘
```

image
: is the 5- to 8-character alphanumeric name of the image you want to clear from the TPF system.

IPL *i*
: is the number of the IPL area you want to clear from the TPF system. The valid range for this number is 1 to the number of IPL areas generated, which can be a maximum of 4.

Note: You can specify this parameter only for the basic subsystem (BSS).

BP
: clears an image that is logically referenced from other images. These other images fall back to their physical copies of the logical references, if they exist.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIMAG HELP
  ZIMAG ?
- If you clear an image from the TPF system, you cannot use that image again until you define it using the ZIMAG DEFINE command. You must also load or copy the image pointer record (CTKX) and all core image restart area (CIMR) components to the image.

Examples

The definition for the IMAGE001 image is cleared from the TPF system in the following example.

```
User:   ZIMAG CLEAR IMAGE001

System: IMAG0006I 19.39.59 IMAGE DEFINITION FOR IMAGE001 CLEARED
```

IPL area 2 is cleared from the TPF system in the following example.

```
User:   ZIMAG CLEAR IPL 2

System: IMAG0026I 19.39.59 IPL AREA 2 CLEARED
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG COPY–Copy Image Components

Use this command to do the following:

- Physically copy the image pointer record (CTKX) from a source image to a target image.
- Physically or logically copy core image restart area (CIMR) components from a source image to a target image. Dump override tables can also be copied along with CIMR components. See the last bullet in “Requirements and Restrictions” for more information.
- Physically copy initial program load (IPL) areas or program areas from a source image to a target image.

If you perform a *logical copy*, the target image simply references the components in the source image. If you perform a *physical copy*, the components are actually copied to the target image and there is no reference to the components in the source image.

Requirements and Restrictions

- The target image must be disabled.
- When copying IPL areas or program areas:
  - The source image must be enabled.
  - The target image cannot be referenced by an enabled image.
- The contents of the program areas are not validated before they are copied. In addition, the compatibility of the programs with the IPAT on the target image is not verified.
- You can copy CIMR components to a target image only after you load or copy CTKX to that target image.
- You can copy CIMR components from the source image only if the components exist as physical copies in the source image, not logical copies.
- You can physically copy CIMR components from the source image only if the sizes of the components in the source image are smaller than or equal to the allocated sizes of the components in the target image.
- You cannot copy the CPS0, ACPL, ICDF and USR1 components in a nonbasic subsystem.
- You must specify both the COMPONENTS ALL and PHYSICAL parameters to copy dump override tables for all genned processors.

Format

```
►►──ZIMAG COpy── ──simage── ──timage── ─────────────────────────────────────►

►──┬─CTKX─────────────────────────────────────────────────┬──────────────────►◄
   │                   ┌─.───────┐                        │
   │                   ▼         │                        │
   ├─COmponents── ──┬─component─┴──┬── ──┬─LOGical──┬─────┤
   │                └─ALL──────────┘     └─PHYsical─┘     │
   ├─IPL──────────────────────────────────────────────────┤
   └─PROGram──┬──────────┬────────────────────────────────┘
              └── ──BP───┘
```

simage
: is the 5- to 8-character alphanumeric name of the source image.

timage
: is the 5- to 8-character alphanumeric name of the target image.

CTKX
: physically copies CTKX from the source image to the target image. All existing logical references to CIMR components are preserved.

COmponents
: copies one or more CIMR components from the source image to the target image.

component
: is the name of a CIMR component that you want to copy to the target image. Valid components in the basic subsystem (BSS) are: CPS0, FCTB, RIAT, SIGT, ICDF, ACPL, IPAT, USR1, and USR2. Valid components in nonbasic subsystems are: FCTB, RIAT, SIGT, IPAT, and USR2.

ALL
: copies all the CIMR components from the source image to the target image. Keep in mind that CIMR components are copied from the source image only if:

- The components are physical copies in the source image, not logical copies.
- The sizes of the components in the source image are smaller than or equal to the allocated sizes of the components in the target image.
- This parameter is specified along with the PHYSICAL parameter. When both the ALL and PHYSICAL parameters are specified the dump override tables are copied from the source image to the target image.

LOGical
: logically copies CIMR components from the source image to the target image.

PHYsical
: physically copies CIMR components from the source image to the target image. Physical copies overwrite any existing logical copies in the target image. If this parameter is specified along with the COMPONENTS ALL parameter, the dump override tables are copied from the source image to the target image.

IPL
: physically copies the IPL areas from the source image to the target image.

PROGram
physically copies the program areas from the source image to the target image. Program version (PVR) records and ADATA file (APRG) records are also copied if they exist on both the source image and the target image.

Note: The E-type loader structures are initialized on the target image when the program areas are copied from the source image.

BP
allows you to copy program areas when the target image has fewer program records allocated than the source image.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

CTKX is physically copied from the IMAGE001 image to the IMAGE003 image in the following example.

```
User:   ZIMAG COPY IMAGE001 IMAGE003 CTKX

System: IMAG0007I 19.39.59 COPY FROM IMAGE001 TO IMAGE003 COMPLETE
```

The CPS0 and RIAT CIMR components are logically copied from the IMAGE001 image to the IMAGE003 image in the following example.

```
User:   ZIMAG COPY IMAGE001 IMAGE003 COMP CPS0.RIAT LOGICAL

System: IMAG0007I 19.39.59 COPY FROM IMAGE001 TO IMAGE003 COMPLETE
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG DEFINE–Define a TPF Image

Use this command to define an *empty* image or to redefine an existing image.

You define an image by associating an initial program load (IPL) area and a program area with the image. You redefine an image by changing the IPL area, program area, or both areas that are already associated with the image.

Note: A core image restart area (CIMR) is also associated with the image. You do not have to specify the CIMR because there is a one-to-one correspondence between images and CIMR areas.

Requirements and Restrictions

You must disable an existing image before you can redefine it. Use the ZIMAG DISABLE command to disable the image.

Format

```
►►──ZIMAG DEFine── ──image── ──Number── ──n── ──IPL── ──i── ──PROGram── ──p──────►◄
```

image
is the 5- to 8-character alphanumeric name of the image. TPF01 is a reserved name used only for image 1.

Note: When redefining an image you must use the existing image name.

Number *n*
is the image number (or CIMR area) you want to associate with the image. The valid range for this number is 1–8. All the image numbers may not be available, depending on your system configuration.

IPL *i*
is the initial program load (IPL) area number you want to associate with the image. The valid range for this number is 1–4. All the IPL area numbers may not be available, depending on your system configuration.

Note: You can specify this parameter only for the basic subsystem (BSS).

PROGram *p*
is the program area number you want to associate with the image. The valid range for this number is 1–8. All the program area numbers may not be available, depending on your system configuration.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIMAG HELP
  ZIMAG ?
- When redefining an image, you only need to specify the IPL parameter, the PROGRAM parameter, or both.

Examples

In the following example, the IMAGEABC image is defined as image number 1, which consists of IPL area 2, program area 5, and CIMR area 1.

```
User:   ZIMAG DEF IMAGEABC NUM 1 IPL 2 PROG 5

System: IMAG0001I 19.39.59 IMAGE 1 DEFINED
        IMAGE NAME IMAGEABC
        IPL AREA 2
        PROGRAM AREA 5
```

In the following example, the IMAGEABC image is redefined to use IPL area 3. The program area remains the same.

```
User:   ZIMAG DEF IMAGEABC NUM 1 IPL 3

System: IMAG0002I 19.39.59 IMAGE 1 REDEFINED
        IMAGE NAME IMAGEABC
        IPL AREA 3 - WAS 2
        PROGRAM AREA 5 - WAS 5
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG DISABLE–Disable a TPF Image

Use this command to disable an image. You cannot perform an initial program load (IPL) on a disabled image.

Requirements and Restrictions

- You can disable only an enabled image; that is, you cannot disable an image that is active.
- You cannot disable the primary image.
- You cannot disable an image if any of the core image restart area (CIMR) components on that image are logically referenced by another enabled image. Use the ZIMAG COPY or ZIMAG MAKEPHYS command to physically copy the logically referenced CIMR components to the enabled image, or use the ZIMAG UNREF command to delete the logical references to the CIMR components before you enter the ZIMAG DISABLE command.

Format

```
►►──ZIMAG DISAble── ──image──────────────────────────────────────►◄
```

image
: is the 5- to 8-character alphanumeric name of the image you want to disable.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

The IMAGE001 image is disabled in the following example.

```
User:   ZIMAG DISABLE IMAGE001

System: IMAG0004I 19.39.59 IMAGE IMAGE001 DISABLED
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG DISPLAY–Display TPF Image

Use this command to display information about online images. The information is obtained from the image control record (ICR).

Requirements and Restrictions

None.

Format

```
►►──ZIMAG DISPlay── ──────────────────────────────────────────────────►

►──┬─IMage── ──image──────────────────────────────┬──────────────►◄
   ├─PROGram──────────────────────────────────────┤
   ├─IPL──────────────────────────────────────────┤
   ├─PROCessor────────────────────────────────────┤
   ├─ALL──────────────────────────────────────────┤
   └─Keypt── ──┬─Active──────────┬── ──CPU── ──┬─id──┬┘
               ├─Backup──────────┤             └─ALL─┘
               └─Staging── ──image┘
```

IMage *image*
: displays the following information about the specified image, where *image* is the 5- to 8-character alphanumeric name of the image:
  - Status of the image
  - Initial program load (IPL) area associated with the image, if the image is in the basic subsystem (BSS)
  - Program area associated with the image
  - CTKX version code
  - For each CIMR component:
    - Version code, if the component is physically loaded
    - Date and time the component was physically loaded
    - Name of the image that is referenced, if the component is a logical copy
    - Names of other images that logically reference the component, if the component is a physical copy.

  Note: If both a physical and logical copy of a component exist on an image, the logical copy is used.

PROGram
: displays all the program areas defined in the system and the images with which they are associated, if any. All of the program areas (1–8) are displayed, even if they are not generated.

IPL
: displays information about all the IPL areas defined in the system and the images with which they are associated, if any. Information about IPLA and IPLB for each IPL area is also displayed. All IPL areas (1–4) are displayed, even if they are not generated.

  Note: The IPL parameter is valid only for the basic subsystem (BSS).

PROCessor
displays the name of the image associated with each processor in the complex and the status of the processor.

ALL
displays the following information about each image defined in the system:
- Image name
- Image number
- Status
- IPL areas associated with the image, if the image is in the basic subsystem (BSS)
- Program areas associated with the image.

For more detailed information about a particular image, specify the IMAGE parameter.

Keypt
displays the following information about working, backup, or staging keypoints:
- Name and version of the keypoint
- Processor ID, if the keypoint is processor unique
- Date and time the keypoint was loaded using the general file loader or the auxiliary loader
- Keypoint patch indicator
- File address of the keypoint.

Note: Keypoint patch records can exist only in the staging or backup areas. The working area always has a complete copy of a keypoint, not a patch.

Active
displays information about keypoints in the working keypoint area.

Backup
displays information about keypoints in the keypoint backup area.

Staging *image*
displays information about the keypoints in the keypoint staging area for the specified image, where *image* is the 5- to 8-character alphanumeric name of the image.

CPU *id*
displays information about all shared keypoints and unique keypoints for the specified processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

CPU ALL
displays information about all keypoints for all processors.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIMAG HELP
  ZIMAG ?
- The format of the time and dates displayed in the image information depends on your installation.

Examples

Information about all the program areas defined in the system is displayed in the following example.

```
User:   ZIMAG DISPLAY PROG

System: IMAG0015I 17.25.51 PROGRAM AREA DISPLAY
        PROG AREA     IMAGE NAME
        PROG1         TPF01
                      VEQUALV
        PROG2
        PROG3
        PROG4
        PROG5
        PROG6
        PROG7
        PROG8
        END OF ZIMAG DISPLAY
```

Information about all the IPL areas defined in the system is displayed in the following example.

```
User:   ZIMAG DISPLAY IPL

System: IMAG0016I 17.25.51 IPL AREA DISPLAY
        IPL   IPLA                   IPLB
        AREA  VC     DATE    TIME     VC     DATE    TIME     IMAGE NAME
        IPL1  40   03/17/94 08.12.45  40   03/17/94 08.12.45  TPF01
        IPL2  40   03/19/94 12.26.38  BS   03/19/94 12.26.38  VEQUALV
        IPL3
        IPL4
        END OF ZIMAG DISPLAY
```

Information about all the images defined in the system is displayed in the following example.

```
User:   ZIMAG DISPLAY ALL

System: IMAG0017I 17.25.51 IMAGE STATUS DISPLAY
        IMAGE NAME  NUM  STATUS    IPL   PROG
        TPF01        1   PRIMARY   IPL1  PROG1
        VEQUALV      2   DISABLED  IPL1  PROG1
                     3   EMPTY
                     4   EMPTY
                     5   EMPTY
                     6   EMPTY
                     7   EMPTY
                     8   EMPTY
        END OF ZIMAG DISPLAY
```

The name of the image associated with each processor in the system is displayed in the following example.

```
User:   ZIMAG DISPLAY PROC

System: IMAG0018I 19.39.59 PROCESSOR STATUS DISPLAY
        CPU  IMAGE NAME  STATUS
         A   IMAGE2      ACTIVE
         B   IMGTHREE    INACTIVE
         C   IMAGE001    ACTIVE
         D   IMAGE001    ACTIVE
         E   IMAGE001    ACTIVE
        END OF ZIMAG DISPLAY
```

Information about the VEQUALV image is displayed in the following example. Notice that VEQUALV has both a physical and logical copy of CSP0 and RIAT. In this case, VEQUALV uses the logical copies.

```
User:   ZIMAG DISP IMAGE VEQUALV

System: IMAG0019I 17.25.51 IMAGE DISPLAY
        NAME - VEQUALV   STATUS - DISABLED  IPL - 1  PROG - 1  CTKX - 40
        COMP    PHYSICAL COPY           LOGICAL        LOGICAL
                VC     DATE    TIME     REF TO IMAGE   REF FROM IMAGE
        FCTB                            TPF01
        CPS0    40   04/28/94 09.26.56  TPF01
        ICDF                            TPF01
        ACPL                            TPF01
        SIGT                            TPF01
        RIAT    40   04/28/94 09.26.56  TPF01
        IPAT                            TPF01
        USR1                            TPF01
        USR2                            TPF01
        END OF ZIMAG DISPLAY
```

Information about all the keypoints loaded to the keypoint staging area for the IMAGE001 image is displayed in the following example.

```
User:   ZIMAG DISPLAY KEYPT STAGING VEQUALV CPU ALL

System: IMAG0020I 17.25.04 KEYPOINT STATUS DISPLAY
        NAME/VERSION  PROC ID      DATE    TIME   PATCH   FILE ADDRESS
        CTKA 40       B          04/28/94 09.47.48  N      F4043805
        CTKB 40       B          04/28/94 09.47.48  N      F4043809
        CTKV 40       SHARED     04/28/94 09.47.48  N      F404380D
        END OF ZIMAG DISPLAY
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG ENABLE–Enable a TPF Image

Use this command to enable an image. An enabled image is one that can be selected for use during an initial program load (IPL).

Requirements and Restrictions

You can enable an image only if all of its CIMR components exist (either physically or logically) and the image uses an IPL area that has both IPLA and IPLB loaded.

Format

```
►►──ZIMAG Enable── ──image──────────────────────────────────────────►◄
```

image
is the 5- to 8-character alphanumeric name of the image you want to enable.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

The IMAG001 image is enabled in the following example.

```
User:   ZIMAG ENABLE IMAGE001

System: IMAG0003I 19.39.59 IMAGE IMAGE001 ENABLED
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG KEYPT–Continue or Abort Keypoint Request

Use this command when prompted by the TPF system to continue or stop the ZIMAG KEYPT MOVE function or the ZIMAG KEYPT RESTORE function.

The TPF system prompts you to enter this command when you need to do the following:

- Verify that you want to replace the keypoints in the working keypoint area.
- Ensure that this processor is in 1052 state, if it is affected by the ZIMAG KEYPT MOVE or RESTORE function.
- Ensure that the other processors affected by the ZIMAG KEYPT MOVE or RESTORE function are stopped (system reset).

Requirements and Restrictions

You have approximately 4 minutes to enter this command from the time you are prompted to do so by the TPF system.

Format

```
►►──ZIMAG Keypt── ──┬─Continue─┬───────────────────────────────────────────►◄
                    └─Abort────┘
```

Continue
continues the ZIMAG KEYPT MOVE or RESTORE function. When you specify this parameter, demand keypointing is stopped.

Abort
stops the ZIMAG KEYPT MOVE or RESTORE function without replacing the keypoints in the working keypoint area.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

The ZIMAG KEYPT MOVE function is continued in the following example.

```
System: IMAG0022I 17.25.26 ENTER - ZIMAG KEYPT CONTINUE - OR - ZIMAG KEYPT ABORT

User:   ZIMAG KEYPT CONT

System: IMAG0012I 17.25.26 CONTINUING WITH - ZIMAG KEYPT MOVE
        IMAG0011I 17.25.26 KEYPOINTS OVERLAID IN THE WORKING AREA
         THE FOLLOWING PROCESSORS MUST BE IPLED -
         PROC-B
         PROC-C
```

The ZIMAG KEYPT RESTORE function is stopped in the following example.

```
System: IMAG0022I 17.25.26 ENTER - ZIMAG KEYPT CONTINUE - OR - ZIMAG KEYPT ABORT

User:   ZIMAG KEYPT ABORT

System: IMAG0170E 17.25.26 ZIMAG KEYPT RESTORE - REQUEST ABORTED
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG KEYPT–Managing Keypoints

Use this command to do the following:

- Move keypoints from a keypoint staging area to the working keypoint area. The original versions of the keypoints in the working keypoint area are copied to the keypoint backup area to ensure that they are not lost if an error occurs.
- Restore keypoints to the working keypoint area from the keypoint backup area.
- Delete keypoints from the keypoint backup area.

Requirements and Restrictions

- Move keypoints to the working keypoint area only from an enabled image. This allows you to perform an initial program load (IPL) on that image, if necessary. If you need to move keypoints to the working keypoint area from a disabled image, use the BP parameter; however, this is not recommended.
- You can enter this command to move or restore keypoints for this processor only if this processor is in 1052 state. All other processors affected by this command must be stopped (system reset).
- You cannot move keypoints to the working keypoint area if a backup copy of the keypoints currently exists in the keypoint backup area. In this case, specify the DELETE parameter to delete the keypoints from the keypoint backup area before you move the keypoints.

Format

```
                                                          ┌─.─────────┐
►►─ZIMAG Keypt──┬─Move──image─┬──KPT──┬─▼─keypoint─┴─┬──CPU────────►
                ├─Restore─────┤       └─ALL──────────┘
                └─Delete──────┘

►──┬─id──┬──┬───────────┬──────────────────────────────────────────►◄
   └─ALL─┘  │      (1)  │
            └──BP───────┘
```

Notes:

1 The BP parameter is valid only when you specify the MOVE parameter.

Move *image*
: replaces keypoints in the working keypoint area with the appropriate keypoints from the keypoint staging area of the specified image, where *image* is the 5- to 8-character alphanumeric name of the image. The original keypoints in the working keypoint area are copied to the keypoint backup area before they are replaced.

Restore
: replaces keypoints in the working keypoint area with the appropriate keypoints from the keypoint backup area.

Delete
: deletes keypoints from the keypoint backup area.

KPT *keypoint*
is the name of a keypoint you want to move, restore, or delete. Valid keypoints are CTK0, CTK1, CTK2, CTK3, CTK4, CTK5, CTK6, CTK9, CTKA, CTKB, CTKC, CTKD, CTKE, CTKI, CTKM, and CTKV.

KPT ALL
moves, restores, or deletes all of the keypoints in the keypoint staging area.

Note: If you specify the CPU *id* parameter, only the processor unique keypoints are moved, restored, or deleted.

CPU *id*
moves, restores, or deletes only processor unique keypoints for the specified processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

CPU ALL
moves, restores, or deletes both processor shared and processor unique keypoints for all processors.

BP
allows you to move keypoints from a keypoint staging area of an image that is disabled.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIMAG HELP
  ZIMAG ?
- If you move or restore a keypoint that has a patch card, only the patch card is moved or restored.
- If you move or restore keypoints, the TPF system prompts you to enter **ZIMAG KEYPT CONTINUE** or **ZIMAG KEYPT ABORT** to continue or end the function.
- Keypoints are loaded to the keypoint staging area using the auxiliary loader. When loading keypoints, the auxiliary loader first clears the keypoint staging area. Therefore, the keypoint staging area always contains only the keypoints that were last loaded to the image. The auxiliary loader does ***not*** clear the keypoint staging area if no keypoints are being loaded.
- The format of the time and dates displayed in the image information depends on your installation.

Examples

Processor shared keypoints CTK3 and CTKV are moved from the keypoint staging area of the VEQUALV image to the working keypoint area in the following example. This example assumes that the command was entered on processor B. All other processors (C in this example) must be stopped before continuing.

```
User:   ZIMAG KEYPT MOVE VEQUALV KPT CTK3.CTKV CPU ALL

System: IMAG0021I 17.25.26 THE FOLLOWING PROCESSORS MUST BE STOPPED -
          PROC-C
        IMAG0022I 17.25.26 ENTER - ZIMAG KEYPT CONTINUE - OR - ZIMAG KEYPT ABORT

User:   ZIMAG KEYPT CONT

System: IMAG0012I 17.25.26 CONTINUING WITH - ZIMAG KEYPT MOVE
        IMAG0011I 17.25.26 KEYPOINTS OVERLAID IN THE WORKING AREA
         THE FOLLOWING PROCESSORS MUST BE IPLED -
         PROC-B
         PROC-C
```

All processor unique keypoints for processor C are moved from the keypoint staging area of the VEQUALV image to the working keypoint area in the following example. This example assumes that the command was entered on processor B.

```
User:   ZIMAG KEYPT MOVE VEQUALV KPT ALL CPU C

System: IMAG0021I 17.25.26 THE FOLLOWING PROCESSORS MUST BE STOPPED -
         PROC-C
        IMAG0022I 17.25.26 ENTER - ZIMAG KEYPT CONTINUE - OR - ZIMAG KEYPT ABORT

User:   ZIMAG KEYPT CONT

System: IMAG0012I 17.25.26 CONTINUING WITH - ZIMAG KEYPT MOVE
        IMAG0011I 17.25.26 KEYPOINTS OVERLAID IN THE WORKING AREA
         THE FOLLOWING PROCESSORS MUST BE IPLED -
         PROC-C
```

Processor unique keypoint CTK1 for processor B is restored from the keypoint backup area to the working keypoint area. This example assumes that the command was entered on processor B.

```
User:   ZIMAG KEYPT REST KPT CTK1 CPU B

System: IMAG0022I 17.25.26 ENTER - ZIMAG KEYPT CONTINUE - OR - ZIMAG KEYPT ABORT

User:   ZIMAG KEYPT CONT

System: IMAG0013I 17.25.26 CONTINUING WITH - ZIMAG KEYPT RESTORE
        IMAG0011I 17.25.26 KEYPOINTS OVERLAID IN THE WORKING AREA
         THE FOLLOWING PROCESSORS MUST BE IPLED -
         PROC-B
```

All processor unique keypoints for processor C are deleted from the keypoint backup area in the following example.

```
User:   ZIMAG KEYPT DELETE KPT ALL CPU C

System: IMAG0014I 19.39.59 KEYPOINTS REMOVED FROM THE BACKUP AREA
```

Processor shared keypoint CTK6 is deleted from the keypoint backup area in the following example.

```
User:   ZIMAG KEYPT DEL KPT CTK6 CPU ALL

System: IMAG0014I 19.39.59 KEYPOINTS REMOVED FROM THE BACKUP AREA
```

Related Information

- See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.
- See the ZTPLD command for more information about the auxiliary loader.

ZIMAG MAKEPHYS–Overlay Logically Referenced Image Components

Use this command to convert logical copies of core image restart area (CIMR) components to physical copies.

Logical copies of CIMR components are converted to physical copies on the *target image*. CIMR components are physically copied to the target image from the *source image*.

Requirements and Restrictions

The size of the physical copy of the CIMR component on the source image must be smaller than or equal to the allocated size of the CIMR component on the target image.

Format

```
                                                    ┌─,──────────┐
                                                    ▼            │
►►──ZIMAG MAKEPHYS── ──timage── ──COmponents── ──┬───component──┴──┬──────────►◄
                                                 └─ALL─────────────┘
```

timage
: is the 5- to 8-character alphanumeric name of the target image where you want to convert a logical copy of a CIMR component to a physical copy.

COmponents
: specifies the logically referenced CIMR components that you want to convert to physical copies.

component
: is the name of a logically referenced CIMR component that you want to convert to a physical copy. Valid components in the basic subsystem (BSS) are: CPS0, FCTB, RIAT, SIGT, ICDF, ACPL, IPAT, USR1, and USR2. Valid components in nonbasic subsystems are: FCTB, RIAT, SIGT, IPAT, and USR2.

ALL
: converts all of the logically referenced CIMR components on the specified target image to physical copies.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIMAG HELP
  ZIMAG ?
- You can enter this command for an image that is either enabled or disabled.

Examples

All the logical copies of CIMR components in the IMAGE001 image are converted to physical copies in the following example.

```
User:   ZIMAG MAKEPHYS IMAGE001 COMP ALL

System: IMAG0008I 19.39.59 PHYSICAL COPY COMPLETE
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG PRIMARY–Set the Primary Image

Use this command to define an image as the primary image. The initial program load (IPL) area of the *primary image* is used during a hard IPL before the system prompts you to select an image.

Requirements and Restrictions

- Exactly one primary image must exist at all times.
- You can define an image as the primary image only if that image is enabled.
- You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZIMAG Primary── ──image──────────────────────────────────────────────►◄
```

image
: is the 5- to 8-character alphanumeric name of the image you want to define as the primary image.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

The IMAGE001 image is defined as the primary image in the following example.

```
User:   ZIMAG PRIMARY IMAGE001

System: IMAG0005I 19.39.59 IMAGE IMAGE001 HAS BEEN DEFINED AS THE PRIMARY IMAGE
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZIMAG UNREF–Delete Logically Referenced Image Components

Use this command to delete logical copies of core image restart area (CIMR) components from an image.

Requirements and Restrictions

The image must be disabled.

Format

```
                                      ┌─.────────┐
                                      ▼          │
►►──ZIMAG Unref── ──image── ──COmponents── ─┬─component─┴─┬──────────────►◄
                                      └─ALL─────────┘
```

image
: is the 5- to 8-character alphanumeric name of the image where you want to delete logical copies of CIMR components.

COmponents
: specifies the logically referenced CIMR components that you want to delete.

component
: is the name of a logically copied CIMR component that you want to delete. Valid components in the basic subsystem (BSS) are: CPS0, FCTB, RIAT, SIGT, ICDF, ACPL, IPAT, USR1, and USR2. Valid components in nonbasic subsystems are: FCTB, RIAT, SIGT, IPAT, and USR2.

ALL
: deletes all the logical copies of CIMR components on the specified image.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZIMAG HELP
ZIMAG ?

Examples

All logical copies of CIMR components in the IMAGE001 image are deleted in the following example.

```
User:   ZIMAG UNREF IMAGE001 COMP ALL

System: IMAG0009I 19.39.59 LOGICAL REFERENCES REMOVED
```

Related Information

See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZINET ADD–Add an Internet Server Application Entry

Use this command to add an Internet server application entry to the Internet daemon configuration file (IDCF).

Requirements and Restrictions

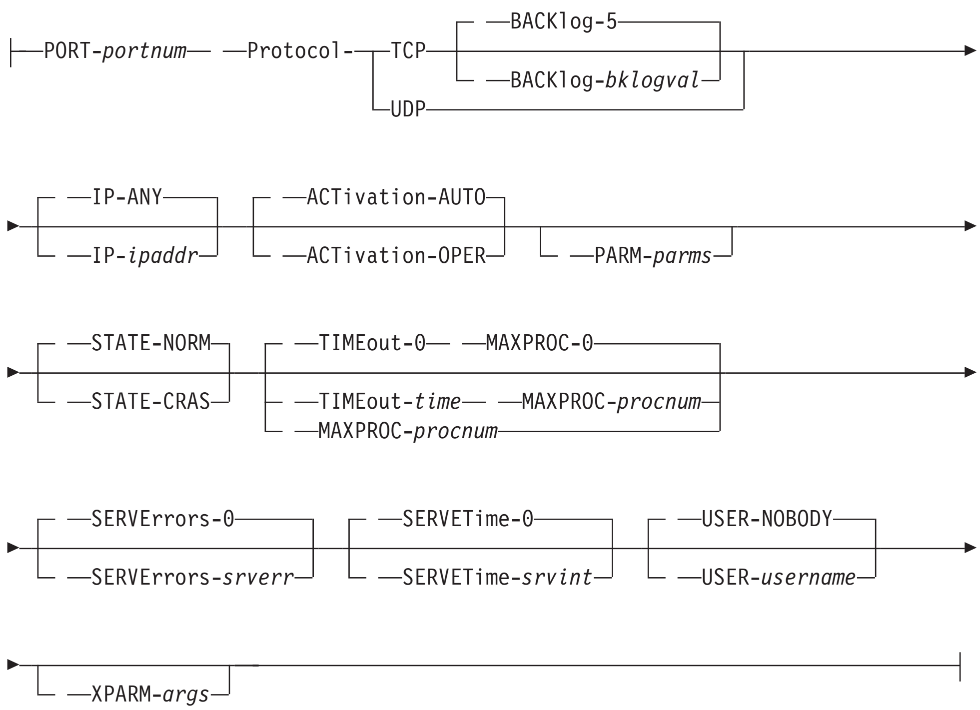
- You can enter this command only in 1052 state or higher.
- You cannot add duplicate Internet server application entries with the same server names on the same processor identifier (ID).
- The IDCF can contain a maximum of 100 Internet server application entries.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - Do not use this command to act on any processor other than the one on which the command is entered. Incomplete data will be saved.
  - If the command is entered on a processor that does not include 32-way loosely coupled processor support, any definitions that are entered must be reentered when 32-way loosely coupled processor support is installed on the processor.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any definitions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor.

Format

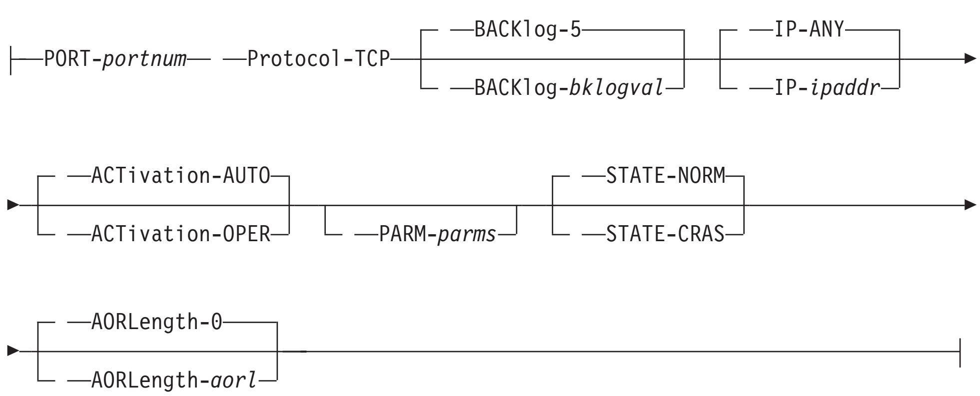
```
►►──ZINET Add── ──Server-sname──┬────────────────────┬── ──PGM-progname──────────►
                                └─ ──PROCid-procid─┘

►─ ──MODEL-──┬─WAIT── ──┤ WAIT and NOWAIT Parameters ├────────┬──────────────►◄
             ├─NOWAIT── ──┤ WAIT and NOWAIT Parameters ├──────┤
             ├─AOR── ──┤ AOR Parameters ├─────────────────────┤
             ├─NOLISTEN── ──┤ NOLISTEN Parameters ├───────────┤
             ├─DAEMON── ──┤ DAEMON Parameters ├───────────────┤
             └─RPC── ──┤ RPC Parameters ├─────────────────────┘
```

WAIT and NOWAIT Parameters:



AOR Parameters:



NOLISTEN Parameters:

```
                                ┌─IP-ANY──────┐ ┌─ACTivation-AUTO─┐
├──┬─────────────────────┬──────┼─────────────┼─┼─────────────────┼────────►
   └─PORT-portnum────────┘      └─IP-ipaddr───┘ └─ACTivation-OPER─┘

                          ┌─STATE-NORM─┐
►──┬─────────────────┬────┼────────────┼──────────────────────────────────┤
   └─PARM-parms──────┘    └─STATE-CRAS─┘
```

DAEMON Parameters:

```
   ┌─ACTivation-AUTO─┐ ┌─STATE-NORM─┐ ┌─TIMEout-0────┐
├──┼─────────────────┼─┼────────────┼─┼──────────────┼──────────────────────►
   └─ACTivation-OPER─┘ └─STATE-CRAS─┘ └─TIMEout-time─┘

   ┌─SERVErrors-0──────┐ ┌─SERVETime-0──────┐ ┌─USER-NOBODY───┐
►──┼───────────────────┼─┼──────────────────┼─┼───────────────┼─────────────►
   └─SERVErrors-srverr─┘ └─SERVETime-srvint─┘ └─USER-username─┘

►──┬─────────────────┬────────────────────────────────────────────────────┤
   └─XPARM-args──────┘
```

RPC Parameters:

```
   ┌─ACTivation-AUTO─┐                  ┌─STATE-NORM─┐
├──┼─────────────────┼─┬──────────────┬─┼────────────┼────────────────────┤
   └─ACTivation-OPER─┘ └─PARM-parms───┘ └─STATE-CRAS─┘
```

Server-*sname*
: specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters.

PROCid-*procid*
: specifies the processor identifier (ID) for which the IDCF entry is being defined, where *procid* is a valid TPF processor ID. If you do not specify this parameter, the processor ID of the processor from which you enter this command is used. If you specify a processor other than the one from which you enter this command, the specified processor ***must*** be inactive.

Note: Do not specify this parameter in a loosely coupled complex where some processors are using a TPF image that includes 32-way loosely coupled processor support and other processors are using an image that does not.

PGM-*progname*
specifies the name of the Internet server application program, where *progname* is a 4-character TPF program name.

MODEL
specifies the process model used by the Internet server application program, where:

WAIT
specifies that the Internet server application is an iterative or single-thread program. The Internet daemon listener starts the Internet server application process with the `tpf_fork` function and will not start any more occurrences of the Internet server application process until the one that is running ends.

NOWAIT
specifies that the Internet server application is a concurrent or a multithread program. The Internet daemon listener starts the Internet server application process with the `tpf_fork` function and will continue to start more occurrences of the Internet server application process until the value specified with the MAXPROC parameter is reached. If the value for the MAXPROC parameter is 0, there is no limit.

AOR
specifies that the Internet server application is a concurrent or a multithread program. The AOR process model can be used only for TCP servers, not UDP servers. The Internet daemon listener starts the Internet server application process with the `activate_on_receipt` function or the `activate_on_receipt_with_length` function if AORLENGTH is nonzero, and will continue to start more occurrences of the Internet server application process until the TPF system cannot handle any more occurrences.

NOLISTEN
specifies that the Internet daemon only starts the Internet server application. The Internet daemon does not create or monitor any sockets, nor does it monitor the server application.

DAEMON
specifies that the Internet server application is an iterative or single-thread program. The Internet daemon listener starts the Internet server application process with the `tpf_fork` function and will not start any more occurrences of the Internet server application process until the one that is running ends. The Internet daemon does not create, bind, or monitor a socket for this process model. Because the `tpf_fork` function returns a process ID, with the DAEMON process model you can stop a process by using the ZINET STOP command. If the child process exits before a ZINET STOP command is entered, the parent process will automatically attempt to activate a new child process.

Note: When using this process model, ensure that your Internet server application uses the `signal` function to enable the SIGTERM signal.

RPC
specifies that the Internet daemon only starts the Internet server application. The Internet daemon does not create or monitor any sockets, nor does it monitor the server application.

PORT-*portnum*
specifies the protocol port number to which the Internet daemon listener will bind the Internet server application socket, where *portnum* is a number from 1 to 65 535.

Protocol
specifies the transport protocol that the Internet server application uses, where:

TCP
specifies Transmission Control Protocol (TCP).

UDP
specifies User Datagram Protocol (UDP).

BACKlog-*bklogval*
specifies the maximum number of connection requests that can be queued to a TCP Internet server application before connection requests are rejected (referred to as the listen backlog value), where *bklogval* is a number from 1 to 32 767.

Note: The maximum listen backlog value for TCP/IP offload support is 5. If the Internet server application is started in a TCP/IP offload environment and the BACKLOG parameter is set to a value greater than 5, the Internet server application will use a value of 5.

IP specifies the Internet Protocol (IP) address to which the Internet daemon listener binds the Internet server application socket, where:

ANY
binds the Internet server application socket to INADDR_ANY.

ipaddr
is the numeric IP address. To specify more local IP addresses for an Internet server application, use the ZINET ALTER command with the ADDIP parameter. An Internet server application can have a maximum of 20 local IP addresses. The Internet daemon will create and monitor a socket for each IP address.

ACTivation
specifies how the Internet server application is started, where:

AUTO
instructs the Internet daemon to automatically start the Internet server application when the Internet daemon is started.

OPER
requires you to enter the ZINET START command to start the Internet server application. The Internet daemon monitor will not create the Internet daemon listener for the Internet server application until you enter a ZINET START command.

PARM-*parms*
specifies a parameter string to be passed to the Internet server application, where *parms* is a 1- to 8-alphanumeric character string.

STATE
specifies the lowest TPF system state in which the Internet server application is allowed to run, where:

NORM
specifies NORM state.

CRAS
specifies CRAS state.

The Internet daemon does not start the Internet server application until the TPF system reaches the specified state. If the TPF system cycles below the specified state, the Internet daemon will stop the Internet server application.

AORLength-*aorl*
specifies the value of the length parameter to be used with the `activate_on_receipt_with_length` function call, where *aorl* is a number from 0 to 32 767. This parameter only has meaning when the model type is AOR. If a value of zero is specified or the default is taken the, `activate_on_receipt` function is used.

TIMEout-*time*
specifies the amount of time, in seconds, the Internet daemon listener will allow an Internet server application process instance to exist, where *time* is a number from 0 to 32 000.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not restrict the time it takes an Internet server application process instance to run.

If you specify a value greater than 0, you must enable SIGTERM signals in your Internet server application because if a timeout occurs, the Internet daemon sends a SIGTERM signal to the Internet server application. Use the `signal` function to enable signals in your applications. See *TPF C/C++ Language Support User's Guide* for more information about the `signal` function.

MAXPROC-*procnum*
specifies the maximum number of Internet server application process instances for every IP address that the Internet daemon listener allows in the TPF system, where *procnum* is a number from 0 to 1000. If the Internet server application is defined with IP-ANY, the value specified for the MAXPROC parameter is the limit. If the Internet server application is defined with more than one IP address, the value specified for the MAXPROC parameter applies to each IP address.

The MAXPROC parameter is useful only when you specify NOWAIT for the MODEL parameter. If you specify WAIT for the MODEL parameter, the Internet daemon listener will use a value of 1 for MAXPROC.

If you specify NOWAIT for the MODEL parameter, and you do not specify the MAXPROC parameter or you specify a value of 0, the Internet daemon listener will use a value of 1000.

SERVErrors-*srverr*
specifies the number of Internet server application process instances that can end in error before the Internet daemon listener stops the Internet server application, where *number* is a number from 0 to 32 000.

This is the number of times the Internet daemon listener will allow the Internet server application to end in error during the sampling interval specified by the SERVETIME parameter. If the error rate is exceeded, the Internet daemon listener stops the Internet server application. Once the Internet server application has been stopped, you must enter the ZINET START command to start the Internet server application again.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not maintain an error rate threshold for the Internet server application.

SERVETime-*srvint*
specifies the sampling time interval, in seconds, that the Internet daemon

listener uses to determine the Internet server application error rate threshold, where *interval* is a number from 0 to 32 000.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not maintain an error rate threshold for the Internet server application.

USER-*username*
specifies the user name of the Internet server application, where *username* is a 1- to 20-character alphanumeric name. You must specify one of the TPF-supplied user names listed in the *TPF C/C++ Language Support User's Guide* for this parameter.

The Internet daemon uses the specified user name to retrieve information about the Internet server application from the password file. The Internet server application is started with the user ID, group ID, and working directory information found in the password file entry.

Note: The user name entered is converted to lowercase; for example, if you enter USER-HTTP, the user name will be stored and displayed as http.

XPARM-*args*
passes a string of parameter data to the **argv** parameter of the `main` function defined in the specified Internet server application program, where *args* is the string of parameter data. The string is case sensitive and can contain any alphanumeric or special characters. End the string with a new-line character (the Enter key).

If you specify this parameter, it ***must*** be the last parameter in the command entry and you ***must*** specify a string of parameter data; specifying a NULL string will cause problems when starting the Internet server application.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZINET HELP
  ZINET ?
- To accommodate long ZINET ADD command entries, you can specify the optional parameters by using the ZINET ALTER command.
- You must enter the ZINET START command to start the Internet server applications added after the Internet daemon has been started. This is true even if the new entry is added with ACTIVATION=AUTO.

Examples

The following example defines the Trivial File Transfer Protocol (TFTP) server application to the IDCF.

```
User:   ZINET ADD S-TFTP PGM-CTFT MODEL-WAIT PORT-69 P-UDP IP-9.117.198.55

System: INET0011I 10:40:12 SERVER TFTP ADDED TO THE
                           INETD CONFIGURATION FILE
```

The following example defines the Apache (HTTP) server application to the IDCF. The -f argument will be passed to the QZZ8 program in the **argv** parameter of the `main` function.

```
User:   ZINET ADD S-APACHE PGM-QZZ8 MODEL-DAEMON USER-root XPARM--f

System: INET0011I 11:22:10 SERVER APACHE ADDED TO THE
                           INETD CONFIGURATION FILE
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.
- See the *TPF C/C++ Language Support User's Guide* for more information about the `tpf_fork` function.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF and about the `activate_on_receipt` function.

ZINET ALTER–Change an Internet Server Application Entry

Use this command to change an Internet server application entry in the Internet daemon configuration file (IDCF).

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - Do not use this command to act on any processor other than the one on which the command is entered. Incomplete data will be saved.
  - If the command is entered on a processor that does not include 32-way loosely coupled processor support, any definitions that are entered must be reentered when 32-way loosely coupled processor support is installed on the processor.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any definitions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor.

Format

```
►►─ZINET ALTer── ──Server-sname─┬──────────────────┬─┬───────────────────┬──►
                                └──PROCid-procid───┘ └──PGM-progname─────┘

►─┬──────────────────┬─┬─────────────────────┬──────────────────────────────►
  └──PORT-portnum────┘ └──Protocol-──┬─TCP─┬─┘
                                     └─UDP─┘

►─┬──────────────────────────┬─┬──────────────────┬─┬──────────────────┬────►
  │     (1)                  │ └──ADDIP-ipaddr────┘ └──DELIP-ipaddr────┘
  └────────BACKlog-bklogval──┘

►─┬──────────────────────────┬─┬─────────────────┬─┬─────────────────┬──────►
  └──ACTivation-──┬─AUTO─┬───┘ └──PARM-parms─────┘ └──STATE-state────┘
                  └─OPER─┘

►─┬──────────────────┬─┬──────────────────────┬─┬────────────────────────┬──►
  └──TIMEout-time────┘ └──MAXPROC-procnum─────┘ └──SERVErrors-number─────┘

►─┬─────────────────────────┬─┬───────────────────┬─┬──────────────────┬────►
  └──SERVETime-interval─────┘ └──USER-username────┘ └──AORLength-aorl──┘

►─┬─────────────────┬───────────────────────────────────────────────────────►◄
  └──XPARM-args─────┘
```

Notes:

1 This parameter is valid only for Internet server applications that use the transport protocol TCP.

Server-*sname*
: specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters.

PROCid-*procid*
: specifies the processor identifier (ID) for which the IDCF entry is being defined, where *procid* is a valid TPF processor ID. If you do not specify this parameter, the processor ID of the processor from which you enter this command is used. If you specify a processor other than the one from which you enter this command, the specified processor ***must*** be inactive.

Note: Do not specify this parameter in a loosely coupled complex where some processors are using a TPF image that includes 32-way loosely coupled processor support and other processors are using an image that does not.

PGM-*progname*
: specifies the name of the Internet server application program, where *progname* is a 4-character TPF program name.

PORT-*portnum*
specifies the protocol port number to which the Internet daemon listener will bind the Internet server application socket, where *portnum* is a number from 1 to 65 535.

Protocol
specifies the transport protocol that the Internet server application uses, where:

TCP
specifies Transmission Control Protocol (TCP).

UDP
specifies User Datagram Protocol (UDP).

BACKlog-*bklogval*
specifies the maximum number of connection requests that can be queued to a TCP Internet server application before connection requests are rejected (referred to as the listen backlog value), where *bklogval* is a number from 1 to 32 767.

Note: The maximum listen backlog value for TCP/IP offload support is 5. If the Internet server application is started in a TCP/IP offload environment and the BACKLOG parameter is set to a value greater than 5, the Internet server application will use a value of 5.

ADDIP-*ipaddr*
specifies an additional Internet Protocol (IP) address to which the Internet daemon will bind the Internet server application socket, where *ipaddr* is a numeric IP address.

You can add a maximum of 20 IP addresses to an Internet server application entry.

DELIP-*ipaddr*
specifies an IP address to delete from the list of IP addresses to which the Internet daemon will bind the Internet server application sockets, where *ipaddr* is a numeric IP address.

If all the IP addresses for an Internet server application are deleted, the Internet daemon binds the socket to INADDR_ANY.

ACTivation
specifies how the Internet server application is started, where:

AUTO
instructs the Internet daemon to automatically start the Internet server application when the Internet daemon is started.

OPER
requires you to enter the ZINET START command to start the Internet server application. The Internet daemon monitor will not create the Internet daemon listener for the Internet server application until you enter a ZINET START command.

PARM-*parms*
specifies a parameter string to be passed to the Internet server application, where *parms* is a 1- to 8-alphanumeric character string.

STATE
specifies the lowest TPF system state in which the Internet server application is allowed to run, where:

NORM
specifies NORM state.

CRAS
specifies CRAS state.

The Internet daemon does not start the Internet server application until the TPF system reaches the specified state. If the TPF system cycles below the specified state, the Internet daemon will stop the Internet server application.

TIMEout-*time*
specifies the amount of time, in seconds, the Internet daemon listener will allow an Internet server application process instance to exist, where *time* is a number from 0 to 32 000.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not restrict the time it takes an Internet server application process instance to run.

If you specify a value greater than 0, you must enable SIGTERM signals in your Internet server application because if a timeout occurs, the Internet daemon sends a SIGTERM signal to the Internet server application. Use the `signal` function to enable signals in your applications. See *TPF C/C++ Language Support User's Guide* for more information about the `signal` function.

MAXPROC-*procnum*
specifies the maximum number of Internet server application process instances for every IP address that the Internet daemon listener allows in the TPF system, where *procnum* is a number from 0 to 1000. If the Internet server application is defined with IP-ANY, the value specified for the MAXPROC parameter is the limit. If the Internet server application is defined with more than one IP address, the value specified for the MAXPROC parameter applies to each IP address.

The MAXPROC parameter is useful only when you specify NOWAIT for the MODEL parameter. If you specify WAIT for the MODEL parameter, the Internet daemon listener will use a value of 1 for MAXPROC.

If you specify NOWAIT for the MODEL parameter, and you do not specify the MAXPROC parameter or you specify a value of 0, the Internet daemon listener will use a value of 1000.

SERVErrors-*srverr*
specifies the number of Internet server application process instances that can end in error before the Internet daemon listener stops the Internet server application, where *number* is a number from 0 to 32 000.

This is the number of times the Internet daemon listener will allow the Internet server application to end in error during the sampling interval specified by the SERVETIME parameter. If the error rate is exceeded, the Internet daemon listener stops the Internet server application. Once the Internet server application has been stopped, you must enter the ZINET START command to start the Internet server application again.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not maintain an error rate threshold for the Internet server application.

SERVETime-*srvint*
specifies the sampling time interval, in seconds, that the Internet daemon listener uses to determine the Internet server application error rate threshold, where *interval* is a number from 0 to 32 000.

If you do not specify this parameter or if you specify a value of 0, the Internet daemon listener will not maintain an error rate threshold for the Internet server application.

USER-*username*
specifies the user name of the Internet server application, where *username* is a 1- to 20-character alphanumeric name. You must specify one of the TPF-supplied user names listed in the *TPF C/C++ Language Support User's Guide* for this parameter.

The Internet daemon uses the specified user name to retrieve information about the Internet server application from the password file. The Internet server application is started with the user ID, group ID, and working directory information found in the password file entry.

Note: The user name entered is converted to lowercase; for example, if you enter USER-HTTP, the user name will be stored and displayed as http.

AORLength-*aorl*
specifies the value of the length parameter to be used with the `activate_on_receipt_with_length` function call, where *aorl* is a number from 0 to 32 767. This parameter only has meaning when the model type is AOR. If a value of zero is specified or the default is taken the, `activate_on_receipt` function is used.

XPARM-*args*
passes a string of parameter data to the **argv** parameter of the `main` function defined in the specified Internet server application program, where *args* is the string of parameter data. The string is case sensitive and can contain any alphanumeric or special characters. End the string with a new-line character (the Enter key).

If you specify this parameter, it ***must*** be the last parameter in the command entry and you ***must*** specify a string of parameter data; specifying a NULL string will cause problems when starting the Internet server application.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZINET HELP
  ZINET ?
- Changes specified with the ZINET ALTER command while the Internet server application is running do not take effect until the Internet server application is stopped and restarted.

Examples

The following example changes the Trivial File Transfer Protocol (TFTP) server entry in the IDCF to monitor an error rate threshold for the TFTP server. The Internet daemon will deactivate the TFTP server if it ends in error 10 times in a 60-second sampling interval.

```
User:   ZINET ALT S-TFTP SERVERRORS-10 SERVETIME-60

System: INET0012I 10.06.07 SERVER TFTP       ENTRY UPDATED
        CSMP0097I 10.06.07 CPU-B SS-BSS  SSU-HPN  IS-01
        SERVER - TFTP       PROCID - B    ACTIVATION - OPER
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - 9.117.198.24

        ALTERED TO -

        SERVER - TFTP       PROCID - B    ACTIVATION - OPER
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00010  SERVETIME - 00060  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - 9.117.198.24
        END OF DISPLAY
```

The following example changes the TFTP entry in the IDCF to listen for input over a second IP address (9.117.198.55).

```
User:   ZINET ALTER SERVER-TFTP ADDIP-9.117.198.55

System: INET0012I 10.07.04 SERVER TFTP       ENTRY UPDATED
        CSMP0097I 10.07.04 CPU-B SS-BSS  SSU-HPN  IS-01
        SERVER - TFTP       PROCID - B    ACTIVATION - OPER
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00010  SERVETIME - 00060  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - 9.117.198.24

        ALTERED TO -

        SERVER - TFTP       PROCID - B    ACTIVATION - OPER
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00005  SERVETIME - 00060  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - 9.117.198.24  9.117.198.55
        END OF DISPLAY
```

The following example changes the Apache entry in the IDCF to add an XPARM parameter.

```
User:   ZINET ALTER SERVER-APACHE XPARM--f /usr/local/apache/conf/httpd.conf

System: INET0012I 11.57.04 SERVER APACHE     ENTRY UPDATED
        CSMP0097I 11.57.04 CPU-B SS-BSS  SSU-HPN  IS-01
        SERVER - APACHE     PROCID - B    ACTIVATION - AUTO
        PGM    - QZZ8       PARM   -
        PROTOCOL   -        PORT      - 00000  MODEL - DAEMON
        SERVERRORS - 00000  SERVETIME - 00000  USER  - root
        MAXPROC    - 00001  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - ANY

        ALTERED TO -

        SERVER - APACHE     PROCID - B    ACTIVATION - AUTO
        PGM    - QZZ8       PARM   -
        PROTOCOL   -        PORT      - 00000  MODEL - DAEMON
        SERVERRORS - 00000  SERVETIME - 00000  USER  - root
        MAXPROC    - 00001  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        XPARM      - -f
        IP - ANY
        END OF DISPLAY
```

The following example changes the maximum number of connection requests that can be queued for the File Transfer Protocol (FTP) server entry in the IDCF.

```
User:   ZINET ALTER SERVER-FTP BACKLOG-100

System: INET0012I 13.28.58 SERVER FTP        ENTRY UPDATED
        CSMP0097I 13.28.58 CPU-B SS-BSS  SSU-HPN  IS-01
        SERVER - FTP        PROCID - B    ACTIVATION - AUTO
        PGM    - CFTP       PARM   -
        PROTOCOL   - TCP    PORT      - 00021  MODEL - NOWAIT
        SERVERRORS - 00000  SERVETIME - 00000  USER  - root
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00005
        IP - ANY

        ALTERED TO -

        SERVER - FTP        PROCID - B    ACTIVATION - AUTO
        PGM    - CFTP       PARM   -
        PROTOCOL   - TCP    PORT      - 00021  MODEL - NOWAIT
        SERVERRORS - 00000  SERVETIME - 00000  USER  - root
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00100
        IP - ANY
        END OF DISPLAY
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.
- See the *TPF C/C++ Language Support User's Guide* for more information about the `tpf_fork` function.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF and about the `activate_on_receipt` function.

ZINET DELETE–Delete an Internet Server Application Entry

Use this command to delete an Internet server application entry from the Internet daemon configuration file (IDCF).

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - Do not use this command to act on any processor other than the one on which the command is entered. Incomplete data will be saved.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any deletions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor.

Format

```
►►──ZINET DELete── ──Server-sname──┬──────────────────┬──────────────────►◄
                                   └── ──PROCid-procid─┘
```

Server-*sname*
specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters.

PROCid-*procid*
specifies the processor identifier (ID) for which the IDCF entry is being deleted, where *procid* is a valid TPF processor ID. If you do not specify this parameter, the processor ID of the processor from which you enter this command is used. If you specify a processor other than the one from which you enter this command, the specified processor ***must*** be inactive.

Note: Do not specify this parameter in a loosely coupled complex where some processors are using a TPF image that includes 32-way loosely coupled processor support and other processors are using an image that does not.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZINET HELP
  ZINET ?
- The Internet server application entry will not be deleted until all child processes associated with that server application end.

Examples

The following example deletes the Trivial File Transfer Protocol (TFTP) server entry from the IDCF.

```
User:   ZINET DEL SERVER-TFTP

System: INET0013I 10:41:12 SERVER TFTP ENTRY DELETED
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF.

ZINET DISPLAY–Display Internet Server Application Entries

Use this command to display the status of the Internet daemon or individual Internet server application entries.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZINET Display──┬──────────────────┬──┬──────────────────────┬──────────────►
                   └──Server-sname────┘  └──Copy-──┬─Active─┬───┘
                                                   └─File───┘

►──┬─────────────────────────┬─────────────────────────────────────────────────►◄
   └──CHild-──┬─ALL─────┬────┘
              └─pidnum──┘
```

Server-*sname*
specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters.

Note: Use the asterisk (*) as a wildcard character to specify a group of Internet server application names. For example, to display information about all Internet server application names beginning with WEB, enter:

ZINET DISP S-WEB*

Copy
specifies the copy of the Internet server application entry values you want to display, where:

Active
displays the Internet server application values that the Internet daemon is currently running.

File
displays the Internet server application values found in the IDCF.

CHild
specifies the child process identifier (ID) you want to display, where:

ALL
displays all the child process IDs created by the Internet server application.

pidnum
displays a single child process ID created by the Internet server application.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZINET HELP
ZINET ?

Examples

The following example displays the Trivial File Transfer Protocol (TFTP) server entry from the IDCF.

```
User:  ZINET D SERVER-TFTP COPY-F

System: INET0014I 10.14.14 START OF ZINET DISPLAY OF FILE
        SERVER - TFTP       PROCID - B    ACTIVATION - OPER
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00005  SERVETIME - 00060  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - 9.117.198.24  9.117.198.55
        END OF DISPLAY
```

The following example displays the Apache server entry from the IDCF.

```
User:  ZINET D SERVER-APACHE COPY-F

System: INET0014I 10.16.14 START OF ZINET DISPLAY OF FILE
        SERVER - APACHE     PROCID - B    ACTIVATION - AUTO
        PGM    - QZZ8       PARM   -
        PROTOCOL   -        PORT      - 00000  MODEL - DAEMON
        SERVERRORS - 00000  SERVETIME - 00000  USER  - root
        MAXPROC    - 00001  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        XPARM      - -f /usr/local/apache/conf/httpd.conf
        IP - ANY
        END OF DISPLAY
```

The following example displays all Internet server application entries found in the IDCF.

Note: The output for this display can span multiple screens; use the ZPAGE command to see the additional screens.

```
User:   ZINET D SERVER-* C-F

System: INET0015I 10.16.13 START OF ZINET DISPLAY OF FILE
        SERVER     PROTO PORT  ACT  PGM  IP
        SERV1      TCP   00001 OPER PGM1 ANY
        SERV2      TCP   00002 OPER PGM2 9.117.198.24  9.117.197.59
                                         9.117.197.60
        TFTP       UDP   00069 OPER CTFT 9.117.198.24  9.117.198.55
        HTTPSERV   TCP   00080 AUTO HTTP ANY
        END OF DISPLAY
```

The following example displays the active TFTP server entry, where:

SOCKET
: is the socket descriptor for the IP address on which the Internet daemon is currently listening. The socket descriptor is displayed in decimal and in hexadecimal format as follows:

  *decimal***/***hexadecimal*

  Note: If the socket descriptor is zero and the process model is WAIT, NOWAIT, or AOR, the Internet daemon listener was unable to create and bind a socket for the IP address and port specified, or the server error threshold has been reached for that IP address and the Internet daemon listener has stopped listening on that IP address.

COUNT
is the number of times the Internet daemon has started the TFTP program to process input from the Internet Protocol (IP) address and socket descriptor specified on the left. The count is set to zero when the TFTP server is started.

TOTAL COUNT
is the sum of all the count fields.

```
User:   ZINET D S-TFTP COPY-ACTIVE

System: INET0031I 11.26.38 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - TFTP       PROCID - B    ACTIVATION - AUTO
        PGM    - CTFT       PARM   -
        PROTOCOL   - UDP    PORT      - 00069  MODEL - NOWAIT
        SERVERRORS - 00005  SERVETIME - 00060  USER  - nobody
        MAXPROC    - 00005  TIMEOUT   - 00120  STATE - NORM
        AORLENGTH  - 00000  BACKLOG   - 00000
        IP - ANY             SOCKET - 00000036/00000024 COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
```

The following example displays all the active child process IDs associated with the Apache server, where:

TOTAL CHILD COUNT
is the sum of all the child process IDs created by the Internet server application.

PROCESS ID
is the child process ID.

TIME STAMP
is the time when the child process ID is created.

```
User:   ZINET DISP S-APACHE CH-ALL

System: INET0030I 11.06.49 START OF ZINET DISPLAY OF CHILD PROCESS
        SERVER - APACHE     TOTAL CHILD COUNT: 00001
        PROCESS ID          TIME STAMP
        ----------          ----------
        1133969750          Thu Sep  2 11:06:34  1999
        END OF DISPLAY
```

The following example displays the specified active child process IDs associated with the active Apache server, where:

PROCESS ID
is the child process ID.

TIME STAMP
is the time when the child process ID is created.

```
User:   ZINET DISP S-APACHE CH-1133772875

System: INET0030I 11.06.07 START OF ZINET DISPLAY OF CHILD PROCESS
        PROCESS ID          TIME STAMP
        ----------          ----------
        1133772875          Thu Sep  2 10:58:27  1999
        END OF DISPLAY
```

The following example displays the active Internet server applications with their child count, where:

SERVER
is the active Internet server application.

CHILD COUNT
is the child count associated with each Internet server application.

```
User:   ZINET DISP S-* CH-ALL

System: INET0029I 15.39.48 START OF ZINET DISPLAY OF CHILD PROCESS COUNT
        SERVER              CHILD COUNT
        -------------       -----------
        TFTP                00001
        APACHE              00001
        END OF DISPLAY
```

The following example displays all active Internet server application entries with which the Internet daemon is currently running, where:

SOCKET
is the socket descriptor for the IP address on which the Internet daemon is currently listening. The socket descriptor is displayed in decimal and in hexadecimal format, as follows:

*decimal***/***hexadecimal*

Note: If the socket descriptor is zero and the process model is WAIT, NOWAIT or AOR, the Internet daemon listener was unable to create and bind a socket for the IP address and port specified, or the server error threshold has been reached for that IP address and the Internet daemon listener has stopped listening on that IP address.

COUNT
is the number of times the Internet daemon has started the TFTP program to process input from the IP address and socket descriptor specified on the left. The count is set to zero when the TFTP server is started.

Note: The output for this display can span multiple screens; use the ZPAGE command to see the additional screens.

```
User:   ZINET D SERVER-* C-A

System: INET0032I 11.26.15 START OF ZINET DISPLAY OF ACTIVE SERVERS
        SERVER     PROTO PORT  PGM  IP               SOCKET            COUNT
        TFTP       UDP   00069 CTFT ANY              00000036/00000024 0000000000
        FTP        TCP   00021 CFTP 9.117.198.74     00000037/00000025 0000000000
                                    9.117.198.70     00000000/00000000 0000000000
        APACHE           00000 CHTA ANY              00000038/00000026 0000000000
        END OF DISPLAY
```

The following example displays the status of the Internet daemon.

```
User:   ZINET DISPLAY

System: INET0026I 11:41:12 THE INETD IS ACTIVE
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF.

ZINET START–Start the Internet Daemon

Use this command to start the Internet daemon or an Internet server application.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- The specified Internet server application must be defined to the processor identifier (ID) of the TPF system from which you enter this command.

Format

```
►►──ZINET START── ─┬──────────────┬──────────────────────────────────►◄
                   └─Server-sname─┘
```

Server-*sname*
: specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters. If you do not specify this parameter, the Internet daemon itself is started.

Note: Use the asterisk (*) as a wildcard character to specify a group of Internet server application names. For example, to start all Internet server application names beginning with WEB, enter:

ZINET START S-WEB*

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZINET HELP
ZINET ?

Examples

The following example starts the Internet daemon.

```
User:   ZINET START

System: INET0016I 08:42:22 THE INETD STARTED
```

The following example starts the WEB Internet server application entry.

```
User:   ZINET START SERVER-WEB

System: INET0017I 08:43:31 SERVER WEB STARTED
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF.

ZINET STOP–Stop the Internet Daemon

Use this command to stop the Internet daemon or an Internet server application entry.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZINET STOP── ─┬─────────────┬─────────────────────────────────►◄
                  └─Server-sname─┘
```

Server-*sname*
: specifies the Internet server application name, where *sname* is from 1 to 10 alphanumeric characters. If you do not specify this parameter, the Internet daemon itself is stopped.

Note: Use the asterisk (*) as a wildcard character to specify a group of Internet server application names. For example, to stop all Internet server application names beginning with WEB, enter:

ZINET STOP S-WEB*

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZINET HELP
  ZINET ?
- If the TPF Internet mail servers are running, enter **ZMAIL STOP** to stop the TPF Internet mail servers ***before*** you enter **ZINET STOP** to stop the Internet daemon.

Examples

The following example stops the Internet daemon.

```
User:   ZINET STOP

System: INET0018I 08:42:22 THE INETD STOPPED
```

The following example stops the WEB Internet server application entry.

```
User:   ZINET STOP S-WEB

System: INET0019I 08:42:32 SERVER WEB STOPPED
```

Related Information

- See *TPF Application Programming* for more information about Internet server applications.

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IDCF.

ZINIP–Manage an Individual IP Trace

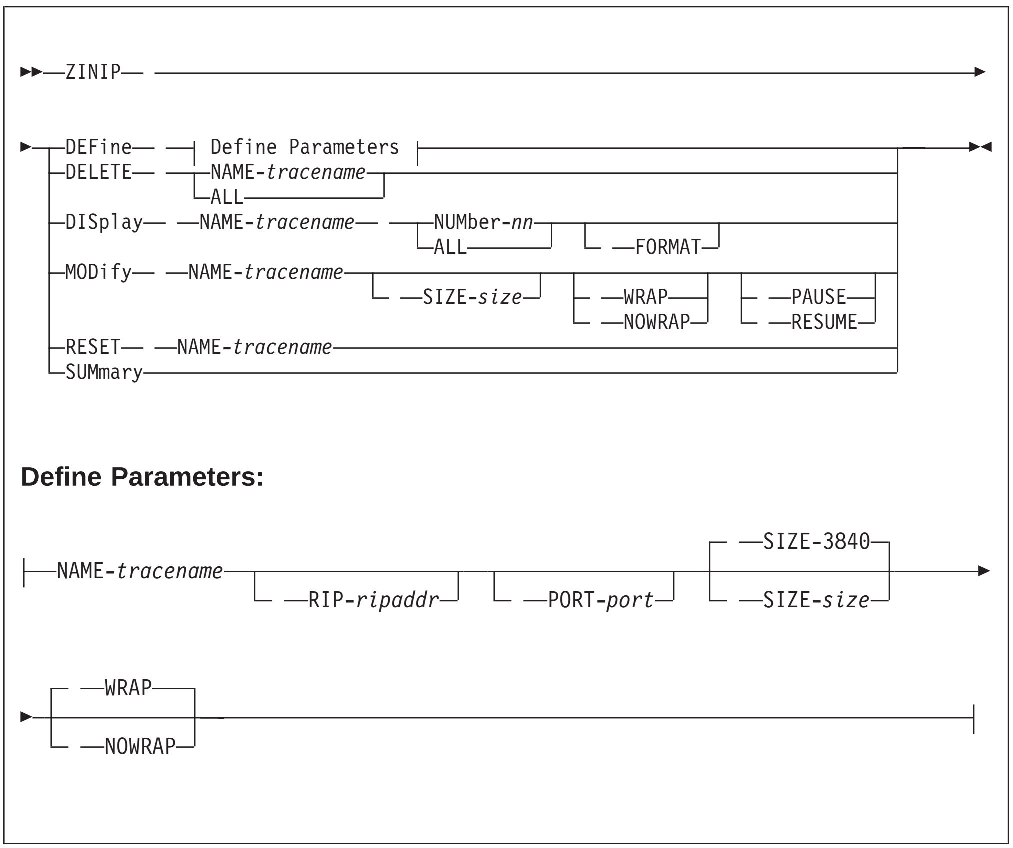
Use this command to do the following:

- Add an entry to the individual Internet Protocol (IP) trace table
- Delete one or all entries from the individual IP trace table
- Display the packets in the individual IP trace
- Modify an entry in the individual IP trace table
- Reset an individual IP trace
- Summarize the entries in the individual IP trace table.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format



DEFine
: creates an entry in the individual IP trace table.

NAME-*tracename*
: specifies an individual trace, where *tracename* is a 1- to 8-character alphanumeric trace name.

RIP-*ripaddr*
: specifies a remote IP address, where *ripaddr* is the numeric IP address. Do not enter the following reserved IP addresses for this parameter:

  - 0.0.0.0

- 255.255.255.255
- 127.0.0.0

Note: Either the RIP or PORT parameter, or both parameters, must be specified.

PORT-*port*
specifies the server port, where *port* is a decimal number from 1 to 65535.

Note: Either the RIP or PORT parameter, or both parameters, must be specified.

SIZE-*size*
changes the amount of data to trace in each packet, where *size* is the amount of data to trace. The value can be from 8 to 3840 bytes.

WRAP
specifies that the individual IP trace table will wrap when it reaches the end.

NOWRAP
specifies that the individual trace table will ***not*** wrap when it reaches the end.

DELETE
deletes one or all entries from the individual IP trace table.

ALL
specifies all the entries in the individual IP trace table or all the packets in an individual IP trace.

DISplay
displays the packets in an individual IP trace.

NUMber-*nn*
specifies the number of packets to display in an individual IP trace, where *nn* is a decimal number from 1 to 999.

FORMAT
provides a formatted display of the packets in the individual IP trace table.

MODIFY
modifies an entry from the individual IP trace table.

PAUSE
pauses the tracing on a specified individual IP trace.

RESUME
resumes the tracing on a specified individual IP trace.

RESET
clears the packets in an individual IP trace.

SUMmary
summarizes the entries in the individual IP trace table.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZINIP HELP
  ZINIP ?
- Definitions can be added or changed in 1052 state or higher. The new or changed definitions can be used immediately (no IPL is required).

- If an exception condition is detected for an input packet, generates an output packet, or both, a reason code is added to the entry in the IP trace table. When you specify the FORMAT parameter, the reason code is displayed if there is one associated with the packet. See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IP trace facility and the specific reason codes that can be displayed.

Examples

The following example displays the last five entries in the IP trace table, where:

RW
: is the read/write operation code.

IN
: is the IP channel command word (IPCCW) area index. X'FF'indicates that you are using local sockets.

SOURCE IP
: is the source IP address.

DEST IP
: is the destination IP address.

SPORT
: is the source port. This field has meaning only for packets using Transmission Control Protocol (TCP) or User Datagram Protocol (UDP).

DPORT
: is the destination port. This field has meaning only for packets using TCP or UDP.

PR
: is the protocol; for example:

  01 specifies Internet Control Message Protocol (ICMP).

  06 specifies TCP.

  11 specifies UDP.

FG
: is the TCP flag byte. This field has meaning only for packets using TCP.

DATA
: is the user data in the IP packet. Only the first 7 bytes of user data are displayed.

```
User:   ZINIP DISPLAY NAME-MARDI NUMBER-5

System: INIP0006I 12.41.57 INDIVIDUAL IP TRACE MARDI DISPLAY
        RW IN    SOURCE IP        DEST IP     SPORT DPORT PR FG DATA
        31 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        32 02   9.117.249.056   9.117.249.058  1025  9999 06 18 4040F1F0F08181
        51 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        52 02   9.117.249.056   9.117.249.058  1025  9999 06 18 4040F1F0F08181
        31 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        639 ENTRIES IN IP TRACE TABLE
```

The following example shows a formatted display of the IP trace table. The information shown in the display depends on the protocol of the IP packet, as follows:

RWI
: is the read/write operation code.

IPCCW
: is the IP channel command word (IPCCW) area index. X'FF'indicates that you are using local sockets.

SOURCE IP
: is the source IP address.

DEST IP
: is the destination IP address.

LEN
: is the length of the IP packet.

TOD
: is the time stamp.

PROTOCOL
: is the protocol of the IP packet, where:

01 specifies ICMP

06 specifies TCP

11 specifies UDP.

SOURCE PORT
: is the source port. This field is only displayed for packets using TCP or UDP.

DEST PORT
: is the destination port. This field is only displayed for packets using TCP or UDP.

SEQ
: is the sequence number in the TCP header. This field is only displayed for packets using TCP.

ACK
: is the acknowledgment number in the TCP header. This field is only displayed for packets using TCP that have the ACK flag set in the TCP flag byte.

WINDOW
: is the window size in the TCP header. This field is only displayed for packets using TCP.

URGENT OFFSET
: is the urgent offset field in the TCP header. If this value is not 0, the packet contains out-of-band (OOB) data. This field is only displayed for packets using TCP.

TCP FLAG BYTE
: is the flag byte in the TCP header. The numeric value is displayed with the name of each bit that is set in the flag byte. This field is only displayed for packets using TCP.

REASON CODE
: is the reason code provided if an exception condition is associated with a packet.

IP HEADER
: is the entire IP header of the packet.

TCP HEADER
: is the entire TCP header of the packet. This field is only displayed for packets using TCP.

UDP HEADER
: is the entire UDP header of the packet. This field is only displayed for packets using UDP.

DATA
: is the user data in the packet that was traced.

```
User:   ZINIP DISPLAY NAME-TEST NUMBER-6 FORMAT

System: INIP0007I 11.23.52 INDIVIDUAL IP FORMATTED TRACE TEST DISPLAY
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-148
          TOD-B6E1240A49A93625  PROTOCOL-01 (ICMP)
          REASON CODE - DISCARDED BY FIREWALL _
          IP HEADER   45000094 15020000 39016F31 0975F119 0975F932
          DATA    0   0    0800EDFB 0A60002B 3C18D608 000AA4E0  .....-.. ..O...u.
                 16  10    292A2B2C                             ....
        RWI-52  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-48 _
          TOD-B6E12411EB6096AE  PROTOCOL-06 (TCP)  SOURCE PORT-1034  DEST PORT-9999
          SEQ-1092329965  WINDOW--1  URGENT OFFSET-0
          TCP FLAG BYTE-02 (SYN)
          IP HEADER   45000030 15040000 39066F8E 0975F119 0975F932
          TCP HEADER  040A270F 411BA1ED 00000000 7002FFFF 78A30000 020405D4 01030304 _
        RWI-51  IPCCW-01  SOURCE IP-9.117.249.50  DEST IP-9.117.241.25  LEN-48
          TOD-B6E12411EBDE212E  PROTOCOL-06 (TCP)  SOURCE PORT-9999  DEST PORT-1034
          SEQ-1092531721  ACK-1092329966  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-12 (ACK, SYN)
          IP HEADER   45000030 1A970000 3C0666FB 0975F932 0975F119 _
          TCP HEADER  270F040A 411EB609 411BA1EE 70122000 616A0000 020405D4 01030304
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-40 _
          TOD-B6E124121F4B0645  PROTOCOL-06 (TCP)  SOURCE PORT-1034  DEST PORT-9999
          SEQ-1092329966  ACK-1092531722  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-10 (ACK)
          IP HEADER   45000028 15050000 39066F95 0975F119 0975F932
          TCP HEADER  040A270F 411BA1EE 411EB60A 50102000 8D520000 _
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-57
          TOD-B6E124121F4B0645  PROTOCOL-06 (TCP)  SOURCE PORT-1034  DEST PORT-9999
          SEQ-1092329966  ACK-1092531722  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-18 (ACK, PSH)
          IP HEADER   45000039 15060000 39066F83 0975F119 0975F932 _
          TCP HEADER  040A270F 411BA1EE 411EB60A 50182000 7BF90000
          DATA    0   0    D7C9D5C7 60D7D6D5 C7000000 01000000  PING-PON G.......
                 16  10    64                                   .
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-140 _
          TOD-B6E124121F4B0645  PROTOCOL-06 (TCP)  SOURCE PORT-1034  DEST PORT-9999
          SEQ-1092329983  ACK-1092531722  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-18 (ACK, PSH)
          IP HEADER   4500008C 15070000 39066F2F 0975F119 0975F932 _
          TCP HEADER  040A270F 411BA1FF 411EB60A 50182000 A3DC0000
          DATA    0   0    4040F1F0 F0818181 81818181 81818181    100aaa aaaaaaaa
                 16  10    81818181                             aaaa
         106 ENTRIES IN IP TRACE TABLE
```

The following example displays a summary of the individual traces defined, where:

NAME
: is the name of the individual trace.

REMOTE IP
: is the remote IP address (if any) being traced.

PORT
: is the server port (if any) being traced.

WRAP
: specifies whether the trace is allowed to wrap.

PAUSED
: specifies whether the trace is currently paused.

STATUS
specifies the status of the trace, where:

EMPTY
specifies that there are no trace entries for this individual trace.

FULL
specifies that the individual trace is currently full and no trace entries can be added because the NOWRAP parameter is set for this trace.

Note: If this field is left blank, the trace is not empty and more trace entries can be added.

```
User:   ZINIP SUMMARY

System: INIP0005I 15.18.29 INDIVIDUAL IP TRACE SUMMARY
        NAME     REMOTE IP       PORT  SIZE WRAP PAUSED STATUS
        -------- --------------  ----- ----  ---- ----- -----
        MQSERV                   1414  3840 YES  YES
        MARDI      9.117.249.058        120 YES  NO     EMPTY
        HTTP       9.117.107.167   80  3840 NO   NO     FULL
        END OF DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IP trace tables and TCP/IP native stack support.

ZIPDB–Refresh the TCP/IP Network Services Database

Use this command to refresh the core copy of the TCP/IP network services database.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZIPDB── ──REFresh────────────────────────────────────────────────────►◄
```

REFresh
refreshes the core copy of the TCP/IP network services database from the `/etc/services` file.

Notes:

1. Changes to the TCP/IP network services database take effect immediately after you enter ZIPDB REFRESH.
2. If you specify the REFRESH parameter and the same application exists in both the previous and updated versions of the `/etc/services` file, the message, byte, and packet counts for that application are not reset to 0 across the refresh operation; therefore, you will not lose any previous statistical data associated with that application.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIPDB HELP
  ZIPDB ?
- To display the TCP/IP network services database, enter the ZFILE cat command for the `/etc/services` file.

Examples

In the following example, the TCP/IP network services database is refreshed from the `/etc/services` file.

```
User:   ZIPDB REFRESH

System: IPDB0001I 10:02:14 NETWORK SERVICES DATABASE REFRESHED
```

Related Information

See the *TPF Transmission Control Protocol/Internet Protocol* for more information about the TCP/IP network services database.

ZIPTR–Display IP Trace Table

Use this command to do the following:

- Display one or more entries in the Internet Protocol (IP) trace table
- Initialize the IP trace table.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZIPTR── ──┬─ALL──┬──────────┬──┬──────────────────────────────────────►◄
              │      └──FORMAT──┘  │
              ├─RESET──────────────┤
              └─num──┬──────────┬──┘
                     └──FORMAT──┘
```

ALL
displays all entries in the IP trace table.

FORMAT
displays formatted entries in the IP trace table.

RESET
initializes the IP trace table.

Note: If you specify this parameter, all IP packets that are being traced may not be written to tape.

num
is a specific number of the most recent entries in the IP trace table that you want to display.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZIPTR HELP
  ZIPTR ?
- If an exception condition is detected for an input packet, generates an output packet, or both, a reason code is added to the entry in the IP trace table. When you specify the FORMAT parameter, the reason code is displayed if there is one associated with the packet. See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IP trace facility and the specific reason codes that can be displayed.

Examples

The following example displays the last five entries in the IP trace table, where:

RW
is the read/write operation code.

IN is the IP channel command word (IPCCW) area index. X'FF'indicates that you are using local sockets.

SOURCE IP
is the source IP address.

DEST IP
is the destination IP address.

SPORT
is the source port. This field has meaning only for packets using the Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) protocol.

DPORT
is the destination port. This field has meaning only for packets using the TCP or UDP protocol.

PR
is the protocol; for example:

01 specifies Internet Control Message Protocol (ICMP).

06 specifies TCP.

11 specifies UDP.

FG
is the TCP flag byte. This field has meaning only for packets using the TCP protocol.

DATA
is the user data in the IP packet. Only the first seven bytes of user data are displayed.

```
User:   ZIPTR 5

System: IPTR0001I 12.41.57 IP TRACE TABLE
        RW IN    SOURCE IP        DEST IP     SPORT DPORT PR FG DATA
        31 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        32 02   9.117.249.056   9.117.249.058  1025  9999 06 18 4040F1F0F08181
        51 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        52 02   9.117.249.056   9.117.249.058  1025  9999 06 18 4040F1F0F08181
        31 02   9.117.249.058   9.117.249.056  9999  1025 06 18 4040F1F0
        639 ENTRIES IN IP TRACE TABLE+
```

The following example shows a formatted display of the IP trace table. The information shown in the display depends on the protocol of the IP packet, as follows:

RWI
is the read/write operation code.

IPCCW
is the IP channel command word (IPCCW) area index. X'FF'indicates that you are using local sockets.

SOURCE IP
is the source IP address.

DEST IP
is the destination IP address.

LEN
is the length of the IP packet.

TOD
: is the time stamp.

PROTOCOL
: is the protocol of the IP packet; for example:

  01 specifies ICMP

  06 specifies TCP

  11 specifies UDP.

SOURCE PORT
: is the source port. This field is displayed only for packets using TCP or UDP.

DEST PORT
: is the destination port. This field is displayed only for packets using TCP or UDP.

SEQ
: is the sequence number in the TCP header. This field is displayed only for packets using TCP.

ACK
: is the acknowledgment number in the TCP header. This field is displayed only for packets using TCP that have the ACK flag set in the TCP flag byte.

WINDOW
: is the window size in the TCP header. This field is displayed only for packets using TCP.

URGENT OFFSET
: is the urgent offset field in the TCP header. If this value is not 0, the packet contains out-of-band (OOB) data. This field is displayed only for packets using TCP.

TCP FLAG BYTE
: is the flag byte in the TCP header. The numeric value is displayed with the names of each bit that is set in the flag byte. This field is displayed only for packets using TCP.

REASON CODE
: is the reason code provided if an exception condition is associated with a packet.

IP HEADER
: is the entire IP header of the packet.

TCP HEADER
: is the entire TCP header of the packet. This field is displayed only for packets using TCP.

UDP HEADER
: is the entire UDP header of the packet. This field is displayed only for packets using UDP.

DATA
: The user data in the packet that was traced.

```
User:   ZIPTR 7 FORMAT

System: IPTR0002I 10.30.59 IP FORMATTED TRACE
        RWI-32  IPCCW-01  SOURCE IP-9.117.107.167  DEST IP-9.117.249.50  LEN-60
          TOD-B6E1183EC53E8646  PROTOCOL-01 (ICMP)
          REASON CODE - REJECTED BY FIREWALL
          IP HEADER   4500003C 76480000 7D014FB5 09756BA7 0975F932 _
          DATA    0   0    0800215C 05002700 61626364 65666768  ........ /.......
                 16  10    696A6B6C                             ..,.
        RWI-31  IPCCW-01  SOURCE IP-9.117.249.50  DEST IP-9.117.107.167  LEN-56
          TOD-B6E1183EC5BF3D06  PROTOCOL-01 (ICMP)
          REASON CODE - REJECTED BY FIREWALL _
          IP HEADER   45000038 D4B20000 3C01324F 0975F932 09756BA7
          DATA    0   0    030AA7A0 00000000 4500003C 76480000  ..x..... ........
                 16  10    7D014FB5                             ....
        RWI-52  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-48
          TOD-B6E118E5B9980844  PROTOCOL-06 (TCP)  SOURCE PORT-1025  DEST PORT-9999
          SEQ-2388309905  WINDOW--1  URGENT OFFSET-0
          TCP FLAG BYTE-02 (SYN)  _
          IP HEADER   45000030 14B60000 39066FDC 0975F119 0975F932
          TCP HEADER  0401270F 8E5AB791 00000000 7002FFFF 15C90000 020405D4 01030304
        RWI-31  IPCCW-01  SOURCE IP-9.117.249.50  DEST IP-9.117.241.25  LEN-48
          TOD-B6E118E5BA17272F  PROTOCOL-06 (TCP)  SOURCE PORT-9999  DEST PORT-1025 _
          SEQ-2388367744  ACK-2388309906  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-12 (ACK, SYN)
          IP HEADER   45000030 D4BF0000 3C06ACD2 0975F932 0975F119
          TCP HEADER  270F0401 8E5B9980 8E5AB792 70122000 CDDB0000 020405D4 01030304
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-40 _
          TOD-B6E118E5EB111744  PROTOCOL-06 (TCP)  SOURCE PORT-1025  DEST PORT-9999
          SEQ-2388309906  ACK-2388367745  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-10 (ACK)
          IP HEADER   45000028 14B70000 39066FE3 0975F119 0975F932
          TCP HEADER  0401270F 8E5AB792 8E5B9981 50102000 F9C30000
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-57
          TOD-B6E118E5EB111744  PROTOCOL-06 (TCP)  SOURCE PORT-1025  DEST PORT-9999
          SEQ-2388309906  ACK-2388367745  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-18 (ACK, PSH)
          IP HEADER   45000039 14B80000 39066FD1 0975F119 0975F932
          TCP HEADER  0401270F 8E5AB792 8E5B9981 50182000 E86A0000
          DATA    0   0    D7C9D5C7 60D7D6D5 C7000000 01000000  PING-PON G....... _
                 16  10    64                                   .
        RWI-32  IPCCW-01  SOURCE IP-9.117.241.25  DEST IP-9.117.249.50  LEN-140
          TOD-B6E118E5EB111744  PROTOCOL-06 (TCP)  SOURCE PORT-1025  DEST PORT-9999
          SEQ-2388309923  ACK-2388367745  WINDOW-8192  URGENT OFFSET-0
          TCP FLAG BYTE-18 (ACK, PSH)  _
          IP HEADER   4500008C 14B90000 39066F7D 0975F119 0975F932
          TCP HEADER  0401270F 8E5AB7A3 8E5B9981 50182000 104E0000
          DATA    0   0    4040F1F0 F0818181 81818181 81818181    100aaa aaaaaaaa
                 16  10    81818181                             aaaa
        1064 ENTRIES IN IP TRACE TABLE
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about the IP trace table and TCP/IP native stack support.

ZIUMP–Initialize Unsolicited Package Records

Use this command to initialize the fixed file records used by the unsolicited message directory (CODR).

Requirements and Restrictions

The TPF system must be in 1052 or UTIL state.

Format

```
►►──ZIUMP── ──┬─ALL──┬──────────────────────────────────────────────────────►◄
              └─CODR─┘
```

ALL
initializes all of the fixed file CODR records.

CODR
initializes all of the fixed file CODR records.

Additional Information

None.

Examples

The fixed file records that are used by the unsolicited message directory are initialized in the following example.

```
User:   ZIUMP CODR

System: IUMP0001I 12.18.20 REQUEST PROCESSED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the unsolicited message processor.

ZKPTR–Keypoint Pointer Record Update

Use this command to update or display the keypoint pointer record during TPF system restart.

Requirements and Restrictions

You can enter this command to update the keypoint pointer record only when prompted during TPF system restart.

Format

```
►►──ZKPTR── ──┬─Replace─────────┬──────────────────────────────►◄
              ├─CAncel──────────┤
              ├─COntinue────────┤
              └─Display─┬─────┬─┘
                        └──n──┘
```

Replace
: replaces the keypoint pointer record that is on file with the pointer record built by system restart.

CAncel
: cancels the TPF system IPL or restart by ending the restart entry control block (ECB) with system error 005100. The existing keypoint pointer record on file is unchanged.

COntinue
: continues the TPF system IPL or restart without filing the keypoint pointer record built by system restart. The existing keypoint pointer record that is on file is unchanged.

  The CONTINUE parameter is primarily intended for use when a new face table (FCTB) that is loaded with expanded keypoint records has been used to update the keypoint pointer records during an IPL and another processor is IPLed using the old FCTB. The CONTINUE parameter allows the processor on the old FCTB to continue processing. You are responsible for making sure that new keypoint records are not accessed until all processors are running on the new FCTB.

Display
: displays the keypoint pointer record. If an update request is outstanding, a comparison of the existing keypoint pointer record on file and the extent table built from the current FACE table (FCTB) is displayed.

n
: is the fallback extent ordinal in the range 0 – 254. The keypoint pointer record for the specified fallback extent is displayed.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZKPTR HELP
ZKPTR ?

Examples

The following example updates the keypoint pointer record on file.

```
System: CVZ60002A KEYPOINT POINTER RECORD OUT OF SYNC
                  ALLOW REPLACEMENT WITH THE UPDATED RECORD
                     ZKPTR REPLACE
                  TO ABORT THE IPL
                     ZKPTR CANCEL
                  TO CONTINUE THE IPL WITHOUT FILING THE RECORD
                     ZKPTR CONTINUE
                  TO DISPLAY THE OUT OF SYNC RECORD
                     ZKPTR DISPLAY

User:   ZKPTR REP

System: KPTR0001I  KEYPOINT POINTER RECORD FILE REQUESTED
        CVZ60004I  KEYPOINT POINTER RECORD FILED
```

In the following example, a keypoint pointer record mismatch with the FCTB is displayed.

```
System: CVZ60002A KEYPOINT POINTER RECORD OUT OF SYNC
                  ALLOW REPLACEMENT WITH THE UPDATED RECORD
                     ZKPTR REPLACE
                  TO ABORT THE IPL
                     ZKPTR CANCEL
                  TO CONTINUE THE IPL WITHOUT FILING THE RECORD
                     ZKPTR CONTINUE
                  TO DISPLAY THE OUT OF SYNC RECORD
                     ZKPTR DISPLAY

User:   ZKPTR D

System: KPTR0020I  #KEYPT   KEYPOINT POINTER RECORD OUT OF SYNC
                             FILE               FCTB
                   EX    CCHHR    K ORD     CCHHR    K ORD
                         FACE ORD  RECS     FACE ORD  RECS
                    0  0029000001     0   0029000001     0
                                0   360            0   360
                    1                     0100000001   361
                                                 720   180
                   *** END OF DISPLAY ***
```

In the following example, a corrupted keypoint pointer record is displayed.

```
System: CVZ60001A KEYPOINT POINTER RECORD CORRUPTED
                  ALLOW REPLACEMENT WITH THE REBUILT RECORD
                     ZKPTR REPLACE
                  TO ABORT THE IPL
                     ZKPTR CANCEL
                  TO DISPLAY THE CORRUPTED RECORD
                     ZKPTR DISPLAY

User:   ZKPTR D

System: KPTR0030I  #KEYPT   KEYPOINT POINTER RECORD CORRUPTED
                   RECORD ID   - X'0000'
                      EXPECTED - X'C3D2'
                   CODE CHECK  - X'00'
                      EXPECTED - X'00'
                   RECORD TYPE - '        ' (X'0000000000000000')
                      EXPECTED - '#KEYPT  '
                   FILED BY    - '    ' (X'00000000')
                   *** END OF DISPLAY ***
```

In the following example, there is no outstanding update request and the keypoint pointer record is displayed.

```
User:   ZKPTR D

System: KPTR0010I  #KEYPT   KEYPOINT POINTER RECORD
                   EX    CCHHR    K ORD  RECS   FACE ORD
                    0  0029000001     0   360          0
                    1  0100000001   360   180        720

                   FALLBACK EXTENTS:
                   EX     CCHHR
                     0  084C000E01
                     1  084E000401
                   *** END OF DISPLAY ***
```

Note: A display of the keypoint pointer record for a fallback extent is similar to the previous example except that fallback extent pointers are not displayed.

Related Information

See *TPF Migration Guide: Program Update Tapes* for more information about 32-way loosely coupled processor support migration.

ZLACL–Alter CCP Restart/Shutdown Polling Controls

Use this command to change the CCP restart and shutdown polling controls. The restart and shutdown polling controls are based on the size of the input list. When the size of the input list exceeds the shutdown level for binary synchronous communication (BSC) lines or 3270 local lines, polling is discontinued. When the size of the input list decreases to below the restart level, polling continues.

Similar control levels are used for ATA/IATA synchronous link control (SLC) communication lines to control the generation of STP ALL (stop all link) control blocks and RSM LCB (resume link) control blocks.

Requirements and Restrictions

None.

Format

```
►►──ZLACL── ──┬─AI─┬── ──restart── ──shutdown──────────────────────────►◄
              ├─BS─┤
              └─LC─┘
```

AI changes the restart and shutdown polling controls for ATA/IATA synchronous link control (SLC) communication lines.

BS
changes the restart and shutdown polling controls for BSC lines.

LC
changes the restart and shutdown polling controls for 3270 local lines.

restart
is the 1- to 3-digit hexadecimal size of the input list needed to continue polling.

shutdown
is the 1- to 3-digit hexadecimal size of the input list that discontinues polling.

Additional Information

- Enter the ZNKEY command to change the SNA restart or shutdown polling controls.
- Enter the ZLDCL command to display the CCP restart and shutdown polling controls.

Examples

The restart and shutdown polling controls for ATA/IATA SLC communication lines are changed in the following example.

```
User:   ZLACL AI 024 028

System: LACL0001I  12.01.26
                RS /SD
          BSC   096/12C
          AI    024/028
          LC    096/12C
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLAEC–Alter CCP Error Counts

Use this command to set the control unit, line, and terminal interchange (TI) error counts that are on file to 0.

Requirements and Restrictions

None.

Format

```
►►──ZLAEC── ──ALL──────────────────────────────────────────────────────────►◄
```

Additional Information

Enter the ZLDLE command to display control unit and line error counts.

Examples

The control unit, line, and TI error counts that are on file are set to 0 in the following example.

```
User:   ZLAEC ALL

System: 12.36.01 LAEC-RESET STARTED
        12.36.01 LAEC-ALL COUNTS RESET OK
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLAPR–Alter BSC Line Priority

Use this command to change the priority of a binary synchronous communication (BSC) point-to-point line.

Requirements and Restrictions

None.

Format

```
►►──ZLAPR── ──BS── ──line── ──┬─PR─┬──────────────────────────────────►◄
                              └─SC─┘
```

line
is a 2-digit hexadecimal symbolic line number.

PR
defines the line as a primary line.

SC
defines the line as a secondary line.

Additional Information

None.

Examples

BSC line 4C is defined as a primary line in the following example.

```
User:   ZLAPR BS 4C PR

System: 12.37.15 LAPR BS 4C OK
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLASL–Alter BSC Send Limit

Use this command to change the send limit of a single selected binary synchronous communication (BSC) line. The *send limit* is the maximum number of messages that can be sent on a line before allowing the other end to send messages.

Requirements and Restrictions

None.

Format

```
►►—ZLASL— —BS— —line— —limit———————————————————————————————►◄
```

line
: is a 2-digit hexadecimal symbolic line number.

limit
: is the 2-digit hexadecimal send limit. A value of X'00' specifies that the line can only receive messages.

Additional Information

None.

Examples

The send limit of BSC line 4C is changed to X'15' in the following example.

```
User:   ZLASL BS 4C 15

System: 12.38.03 LASL BS 4C OK
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLASN–Assign a Control Unit or Line

Use this command to:

- Assign communication lines to a 2703 control unit and then place the control unit in the online state
- Assign a symbolic communication line to a specified symbolic line configuration table (SLCT) path.

Requirements and Restrictions

- Before you can enter this command, you must enter the ZLIDL command for the 2703 control unit or the communication line.
- When you assign communication lines, the 2703 control unit must be in the offline state.

Format

```
►►──ZLASN── ──┬─CU── ──number──────────────────┬──────────────────────►◄
              └──┬─AI─┬── ──line── ──address─┘
                 └─BS─┘
```

CU
assigns communication lines to a 2703 control unit.

AI assigns a symbolic ATA/IATA synchronous link control (SLC) communication line to an SLCT path.

BS
assigns a symbolic binary synchronous communication (BSC) line to an SLCT path.

line
is a 2-digit hexadecimal symbolic line number.

address
is an SLCT path address from X'00'–X'03'.

number
is a symbolic control unit number.

Additional Information

None.

Examples

Control unit 01 is assigned communication lines in the following example.

```
User:   ZLASN CU 01

System: 18.49.30 LASN CU01 ASSIGN COMPLETE
```

BSC line 44 is assigned to SLCT path 00 in the following example.

```
User:   ZLASN BS 44 00

System: 19.02.58 LVAL BS44 SUB6A BS LN VALIDATED
        19.02.58 LSTA BS44 CU00 SUB06A ON OK
        19.02.58 LASN BS44 CU00 CHN00 ADD6A SUB006A  ASSIGN COMPLETE
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLASP–Alter BSC Multipoint Line Slow Poll Interval

Use this command to change the slow poll interval for a specified binary synchronous communication (BSC) multipoint line for which the TPF system is the control station.

The *slow poll interval* is the time interval that the control station sends poll messages to a tributary station that is not responding to the polling.

Requirements and Restrictions

None.

Format

```
►►──ZLASP── ──BS── ──line── ──interval─────────────────────────────────►◄
```

line
is a 2-digit hexadecimal symbolic line number.

interval
is a 2-digit hexadecimal number that represents the time, in seconds, of the desired slow poll interval.

Additional Information

None.

Examples

The slow poll interval for BSC line number 41 is changed to 48 seconds (X'30') in the following example.

```
User:   ZLASP BS 41 30

System: 12.39.40 LASP BS 41 30 OK
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLDCL–Display CCP Restart/Shutdown Polling Controls

Use this command to display the CCP restart and shutdown polling controls. The restart and shutdown polling controls are based on the size of the input list. When the size of the input list exceeds the shutdown level for binary synchronous communication (BSC) lines or 3270 local lines, polling is discontinued. When the size of the input list decreases to below the restart level, polling continues.

Similar control levels are used for ATA/IATA synchronous link control (SLC) communication lines to control the generation of STP ALL (stop all link) control blocks and RSM LCB (resume link) control blocks.

Requirements and Restrictions

None.

Format

```
►►──ZLDCL──────────────────────────────────────────────────────────►◄
```

Additional Information

Enter the ZLACL command to change the CCP restart and shutdown polling counts.

Examples

The CCP restart and shutdown controls are displayed in the following example.

```
User:   ZLDCL

System: LDCL0001I  13.25.52
                RS /SD
          BSC   096/12C
          AI    096/12C
          LC    096/12C
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

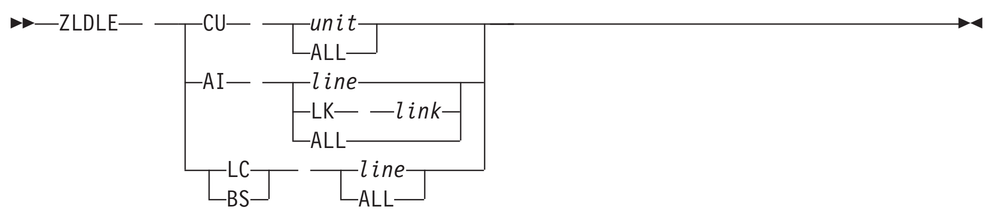
ZLDLE–Display Control Unit and Line Error Counts

Use this command to display the error counts on file for a control unit or communication line.

Requirements and Restrictions

None.

Format



CU
displays the control unit error counts that are on file.

AI displays the ATA/IATA synchronous link control (SLC) communication line errors that are on file.

LC
displays the local 3270 line errors that are on file.

BS
displays the binary synchronous communication (BSC) line errors that are on file.

unit
is a 2-digit hexadecimal control unit number.

ALL
displays all of the control unit or line error counts that are on file.

line
is a 2-digit hexadecimal symbolic line number.

LK
displays the ATA/IATA SLC communication line errors for a specific link.

link
is a 2-digit hexadecimal symbolic link number.

Additional Information

- The line error count includes the line errors that accumulated since the line was assigned. When you assign a new line, the line error count is reset to 0.
- Enter the ZLAEC command to set the error counts to 0.

Examples

The error counts for control unit 00 are displayed in the following example.

```
User:   ZLDLE CU 00

System: 12.41.07 LDLE CU 00 ERRORS 0000
```

The error counts for all the BSC lines are displayed in the following example.

```
User:   ZLDLE BS ALL

System: 12.41.40 LDLE-BS DISPLAY ALL
        BS 3E CU 00 SUB 064     ERRORS 0000 0000
        BS 3F CU 00 SUB 065     ERRORS 0000 0000
        BS 40 CU 00 SUB 066     ERRORS 0000 0000
        BS 41 CU 00 SUB 067     ERRORS 0000 0000
        BS 42 CU 00 SUB 068     ERRORS 0000 0000
        BS 43 CU 00 SUB 069     ERRORS 0000 0000
        BS 44 CU 00 SUB 06A     ERRORS 0000 0000
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLDLS–Display Line Status or Communication Control Unit Status

Use this command to display information about the configuration of the current control unit or to display information about the following communication lines:

- Binary synchronous communication (BSC) lines
- Pseudo lines
- ATA/IATA synchronous link control (SLC) communication lines
- 3270 local lines.

Requirements and Restrictions

None.

Format

```
►►──ZLDLS── ──┬─CU──────────────────────────────┬──────────────────►◄
              ├─AI── ──┬─line──────────┬────────┤
              │        ├─LK── ──link───┤        │
              │        └─ALL───────────┘        │
              ├─┬─LC─┬── ──┬─line─┬─────────────┤
              │ └─BS─┘     └─ALL──┘             │
              └─PS── ──┬─pseudo─┬───────────────┘
                       └─ALL────┘
```

CU
displays information about the configuration of the current communication control unit.

AI displays information about the ATA/IATA SLC communication lines.

LC
displays information about the 3270 local lines.

BS
displays information about the BSC lines.

PS
displays information about the pseudo lines

line
is a 2-digit hexadecimal symbolic line number.

ALL
displays information about all the BSC lines, pseudo lines, local 3270 lines, or ATA/IATA SLC communication lines.

pseudo
is a 2-digit hexadecimal pseudo line number.

LK
displays information about ATA/IATA SLC communication lines for a specific link.

link
is a 2-digit hexadecimal symbolic link number.

Additional Information

None.

Examples

Information about BSC line number 41 is displayed in the following example.

```
User:   ZLDLS BS 41

System: LDLS0000I 12.42.35 BS LINE STATUS
        BS-41 CU-00 SUB-067 SL-65 CNT-0000 INV STP IDL CTL SP-1E
```

Information about the configuration of the current communication control unit is displayed in the following example.

```
User:   ZLDLS CU

System: 12.42.43 LDLS CU CONFIGURATION NOW IS
        CU  TYPE  CHN  STATUS  ADD-RNG
        00  2703   0    ON      60 7F
        01  2703   0    ON      40 4F
```

Information about link number 39 of the ATA/IATA SLC communication lines is displayed in the following example.

```
User:   ZLDLS AI LK 39

System: LDLS0001I 12.45.39 LDLS AI LK 39
        LN 39 SUB 79 78 IND 9080 8080 8000 INV STA
        LN 3A SUB 7B 7A IND 9080 8080 8000 INV STA
        LN 3B SUB 7D 7C IND 9080 8080 8000 INV STA
        LN 3C SUB 00 00 IND 9080 8080 8000 INV STA
        LN 3D SUB 00 00 IND 9080 8080 8000 INV STA
```

Information about 3270 local line number 4E is displayed in the following example.

```
User:   ZLDLS LC 4E

System: LDLS0020I 14.31.26 LC LINE STATUS
        LC4E SDA08B IND2010 VAL STA NBS RCV
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLDTI–Display Terminal Interchange (TI) Status

Use this command to display the status of the stations on a binary synchronous communication (BSC) multipoint line.

Requirements and Restrictions

None.

Format

```
►►──ZLDTI── ──BS── ──line──────────────────────────────────────────►◄
```

line
is a 2-digit hexadecimal symbolic line number.

Additional Information

None.

Examples

The status of the stations on BSC line number 41 is displayed in the following example.

```
User:   ZLDTI BS 41

System: 12.47.55 LDTI BS 41
        ONLINE 00 01 02 03 04 05 06 07 08 09 0A 0B 0C
        OFFLIN NONE
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLIDL–Idle a Control Unit or Communication Line

Use this command to idle all communication lines on a control unit or to idle a specific binary synchronous communication (BSC) line or ATA/IATA synchronous link control (SLC) communication line.

Requirements and Restrictions

To idle a 2703 control unit, you must first place it offline.

Format

```
►►──ZLIDL── ──┬─CU── ──unit──┬───────────────────────────────────►◄
              └─┬─BS─┬── ──line─┘
                └─AI─┘
```

CU
idles all the lines on a control unit.

BS
idles a BSC line.

AI idles an ATA/IATA SLC communication line.

unit
is a 2-digit hexadecimal control unit number.

line
is a 2-digit hexadecimal symbolic line number.

Additional Information

- You do not need to enter the ZLSTP command before you idle a BSC line.
- Communication lines are often idled automatically by the control program. This is indicated by the DOWN flag that is appended to an unsolicited error message. If you notice an excessive rate of I/O errors, it is not necessary to idle the communication line. You can simply enter the ZLSTP command. When you idle or stop a communication line, no more I/O operations are started.

Examples

The communication lines on control unit number 01 are idled in the following example.

```
User:   ZLIDL CU 01

System: 18.51.12 SKN 35 ALL CHANNELS NON-FUNCTIONING - CYCLED DOWN
        18.51.13 LIDL CU01 IDLE COMPLETE
```

BSC line number 44 is idled in the following example.

```
User:   ZLIDL BS 44

System: 10.06.36 LIDL BS44 CU00 CHN00 ADD6A SUB006A  IDLE COMPLETE
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLKST–Display Synchronous Link Statistics

Use this command to display or clear the ATA/IATA synchronous link control (SLC) link counts.

Requirements and Restrictions

None.

Format

```
►►──ZLKST── ──AI── ──┬─line──────────┬──┬───────┬──────────────────────────►◄
                     └─LK── ──link─┘  └─ ──CL─┘
```

line
is a 2-digit hexadecimal symbolic line number.

LK
displays the counts for an SLC link rather than an SLC line.

link
is a 2-digit hexadecimal symbolic link number.

CL
resets the counts to 0.

Additional Information

None.

Examples

The counts for ATA/IATA SLC communication line number 34 are displayed in the following example, where:

ACK
is the acknowledge counts.

ENQ
is the enquiry counts.

AML
is the acknowledge message label counts.

RSM
is the resume sending counts.

```
User:   ZLKST AI 34

System: LKST0001I LKST AI 34
        AI 34
                    I/P      O/P
        ACK           0       58
        ENQ          37       36
        AML           0       15
        RSM          38       37
                     SINGLE    MULTI    CHARS   BLOCKS
        TYPE A I/P       18       22   18,186       91
        TYPE B I/P       14       15   11,450       58
```

The counts for SLC link number 39 are displayed in the following example and then set to 0.

```
User:   ZLKST AI LK 39 CL

System: LKST0001I LKST AI LK 39
        AI 39
                    I/P      O/P
        ACK          59        0
        ENQ          67       68
        AML          15        0
        RSM          67       69
                     SINGLE    MULTI    CHARS   BLOCKS
        TYPE A O/P        2        0      174        2
        TYPE B O/P       14       15   11,450       58
        AI 3A
                    I/P      O/P
        ACK          46        0
        ENQ          72       73
        AML          14        0
        RSM          72       75
                     SINGLE    MULTI    CHARS   BLOCKS
        TYPE A O/P       12       12    6,432       37
        TYPE B O/P        6        2    1,605       10
        AI 3B
        ALL COUNTS ZERO
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLKTF–Stop Synchronous Link Control Link Trace

Use this command to stop the synchronous link control (SLC) link trace and close the LKT general tape.

Requirements and Restrictions

None.

Format

```
►►──ZLKTF──────────────────────────────────────────────────────────────────►◄
```

Additional Information

Enter the ZLKTN command to start the SLC link trace.

Examples

The SLC link trace is stopped and the LKT general tape is closed in the following example.

```
User:   ZLKTF

System: LKTN0002I 17.06.51 LINK TRACE OFF OK.
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about the SLC link trace.

ZLKTN–Start Synchronous Link Control Link Trace

Use this command to start the synchronous link control (SLC) link trace. SLC data blocks and link control blocks (LCBs) for all links are recorded on the LKT general tape.

Requirements and Restrictions

None.

Format

```
►►──ZLKTN──────────────────────────────────────────────────────►◄
```

Additional Information

Enter the ZLKTF command to stop the SLC link trace.

Examples

The SLC link trace is started in the following example.

```
User:   ZLKTN

System: LKTN0001I 17.03.16 LINK TRACE ON OK.
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about the SLC link trace.

ZLREP–Replace a Line or Control Unit

Use this command to do the following:

- Idle a symbolic communication line and then assign it to the first available different symbolic line configuration table (SLCT) path.
- Replace all communication lines on a control unit. The communication lines are idled and then assigned to a new control unit. You can specify the control unit or you can assign the communication lines to the first available control unit.

Requirements and Restrictions

None.

Format

```
►►──ZLREP── ──┬─┬─AI─┬── ──line──────────────┬──────────────────────────────►◄
              │ └─BS─┘                        │
              └─CU── ──unit──┬──────────────┬─┘
                             └── ──symbcu───┘
```

AI replaces an ATA/IATA synchronous link control (SLC) communication line.

BS
replaces a binary synchronous communication (BSC) line.

line
is a 2-digit hexadecimal symbolic line number.

CU
replaces all the communication lines on the specified control unit.

unit
is the 2-digit hexadecimal control unit number of the control unit being replaced.

symbcu
is the 2-digit hexadecimal symbolic control unit number of the control unit to which the communication lines are being assigned. If you do not specify a symbolic control unit number, the communication lines are assigned to the first available control unit.

Additional Information

None.

Examples

BSC line number 44 is replaced in the following example.

```
User:   ZLREP BS 44

System: 10.13.27 LREP BS44 CU00 CHN00 ADD6A SUB006A  IDLE COMPLETE
        10.13.27 LREP BS44 PATH 01 MANUALLY SWITCH
```

The communication lines on control unit 01 are replaced in the following example. The communication lines are idled and then assigned to the first available control unit.

```
User:   ZLREP CU 01

System: 19.06.30 SKN 35 ALL CHANNELS NON-FUNCTIONING - CYCLED DOWN
        19.06.34 SKN 39 ALL CHANNELS NON-FUNCTIONING - CYCLED DOWN
        19.06.41 LREP CU01 REPLACE COMPLETE
```

The communication lines are idled and then assigned to control unit 00 in the following example.

```
User:   ZLREP CU 01 00

System: 10.10.52 LREP CU01 REPLACE COMPLETE
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLRST–Send Terminal Reset

Use this command to send a terminal reset message to a network extension facility (NEF) terminal.

Requirements and Restrictions

You can enter this command only from a CRAS terminal.

Format

```
►►──ZLRST── ──HS── ──pseudo── ──address── ──ta──────────────────────────────►◄
```

pseudo
is a 2-digit hexadecimal NEF pseudo line number.

address
is an NEF pseudo interchange address.

ta
is an NEF pseudo terminal address.

Additional Information

None.

Examples

A terminal reset message is sent to the specified NEF terminal in the following example.

```
User:   ZLRST HS 56 03 04

System: 16.49.59 ------TERMINAL RESET COMPLETE
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLSTA–Start Lines

Use this command to start the following types of communication lines:

- Binary synchronous communication (BSC) lines
- Pseudo communication lines
- ATA/IATA synchronous link control (SLC) communication lines
- 3270 local communication lines.

Requirements and Restrictions

- Enter the ZLVAL command to validate the communication lines before you enter this command.
- If you are starting a BSC line, enter the ZLTON command to validate the station on that line before you enter this command.
- If you are starting a 3270 local line that is interrupt driven, the 3270 local device must be in start mode or interrupts will be ignored.
- This command is valid only in 1052 state, NORM state, or while cycling up to NORM state.
- If the system is in 1052 or UTIL state, only input from 3270 local devices that are logged on to the system message processor is accepted.

Format

```
►►──ZLSTA── ──┬─ALL──────────────────────┬──────────────────────────────►◄
              ├─┬─BS─┬── ──┬─line─┬──────┤
              │ └─LC─┘     └─ALL──┘      │
              ├─PS── ──┬─pseudo─┬──┬───┬─┤
              │        └─ALL────┘  ├─N─┤ │
              │                    └─R─┘ │
              └─AI── ──┬─line────────────┘
                       ├─LK── ──link─┤
                       └─ALL─────────┘
```

ALL
starts all the communication lines.

BS
starts the BSC lines.

LC
starts the 3270 local communication lines.

PS
starts the pseudo lines.

AI starts the ATA/IATA SLC communication lines

line
is a 2-digit hexadecimal symbolic line number.

pseudo
is a 2-digit hexadecimal pseudo line number.

N specifies the normal polling rate.

R specifies the reduced polling rate.

LK
starts the ATA/IATA SLC communication lines on a link.

link
is a 2-digit hexadecimal symbolic link number.

Additional Information

Enter the ZLSTP command to stop communication lines.

Examples

All the communication lines are started in the following example.

```
User:   ZLSTA ALL

System: 13.43.15 LSTA ALL ON OK
```

All the BSC lines are started in the following example.

```
User:   ZLSTA BS ALL

System: 14.16.23 LSTA BS ALL ON OK
```

ATA/IATA SLC communication line number 35 is started in the following example.

```
User:   ZLSTA AI 35

System: 15.07.08 LSTA AI 35    ON OK
```

Related Information

- See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.
- See *TPF Data Communications Services Reference* for more information about the system message processor.

ZLSTP–Stop Lines

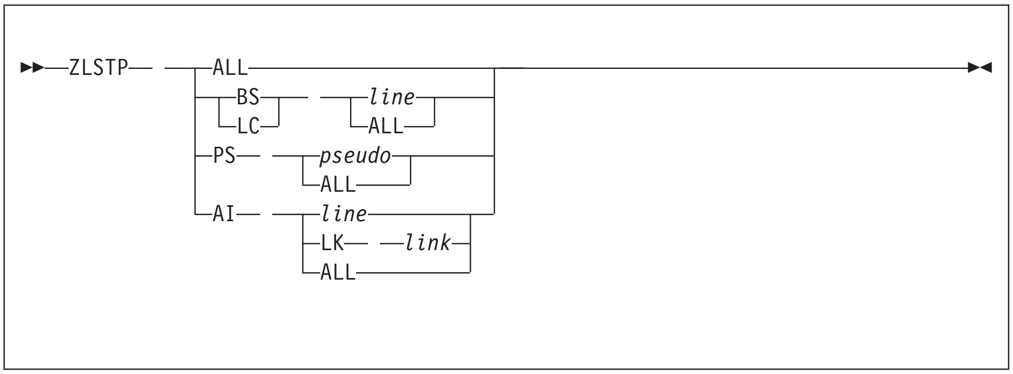
Use this command to stop the following types of communication lines:

- Binary synchronous communication (BSC) lines
- Pseudo communication lines
- ATA/IATA synchronous link control (SLC) communication lines
- 3270 local communication lines.

Requirements and Restrictions

3270 local devices logged on to the system message processor cannot be stopped.

Format



ALL
stops all the communication lines.

BS
stops the BSC lines.

LC
stops the 3270 local communication lines.

PS
stops the pseudo lines.

AI stops the ATA/IATA SLC communication lines

line
is a 2-digit hexadecimal symbolic line number.

pseudo
is a 2-digit hexadecimal pseudo line number.

LK
stops the ATA/IATA SLC communication lines on a link.

link
is a 2-digit hexadecimal symbolic link number.

Additional Information

- Although input is no longer allowed on communication lines that are stopped, the current activity is allowed to be completed.
- Communication lines stop automatically when the TPF system is cycled down.
- For 3270 local lines that are interrupt driven, this command causes interrupts from a device to be ignored.

- 3270 local printers that are used as CRAS printers should be permanently logged on to the system message processor.
- Enter the ZLSTA command to start communication lines.

Examples

All the communication lines are stopped in the following example.

```
User:   ZLSTP ALL

System: 15.29.53 LSTP ALL OFF OK
```

All the BSC lines are stopped in the following example.

```
User:   ZLSTP BS ALL

System: 15.31.05 LSTP BS BEGINS...
        15.31.05 LSTP BS COMPLETE...ALL OFF OK
```

ATA/IATA SLC communication line number 35 is stopped in the following example.

```
User:   ZLSTP AI 35

System: 15.33.10 LSTP AI 35    OFF OK
        15.33.10 SKN 35 LINK DOWN
```

Related Information

- See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.
- See *TPF Data Communications Services Reference* for more information about the system message processor.

ZLTOF–Invalidate a BSC Multipoint Line Station

Use this command to invalidate a station on a binary synchronous communication (BSC) multipoint line.

After the station is invalidated, the TPF system does not poll the station to solicit input messages and does not select the station to transmit output messages.

Requirements and Restrictions

None.

Format

```
►►—ZLTOF— —BS— —line— —station————————————————————————————►◄
```

line
: is a 2-digit hexadecimal symbolic line number.

station
: is the station identification, which is the hexadecimal ordinal number from X'00'–X'3F', of the entry for the station in the BSC station addressing table.

Additional Information

Enter the ZLTON command to validate a BSC multipoint line station.

Examples

Station 01 on BSC line number 44 is invalidated in the following example.

```
User:   ZLTOF BS 44 01

System: 15.36.18 LTOF BS 44 01 OFF
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLTON–Validate a BSC Multipoint Line Station

Use this command to validate a station on a binary synchronous communication (BSC) multipoint line.

After the station is validated, the TPF system polls the station to solicit input messages and selects the station to transmit output messages, provided that the BSC line is started.

Requirements and Restrictions

None.

Format

```
►►──ZLTON── ──BS── ──line── ──station──────────────────────────────────────►◄
```

line
is a 2-digit hexadecimal symbolic line number.

station
is the station identification, which is the hexadecimal ordinal number from 00–3F, of the entry for the station in the BSC station addressing table.

Additional Information

- Enter the ZLTOF command to invalidate a BSC multipoint line station.
- Enter the ZLVAL command to validate communication lines.
- Enter the ZLSTA command to start communication lines.

Examples

Station 01 on BSC line number 44 is validated in the following example.

```
User:   ZLTON BS 44 01

System: 15.36.56 LTON BS 44 01 ON
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZLTRF–CCP Trace Termination

Use this command to force CCP trace into an in-core loop mode trace on just the PRC and RO subchannels. Tracing on all other subchannel addresses is stopped. The final blocks are written to the RTL/RTA tape if necessary.

Requirements and Restrictions

None.

Format

```
►►──ZLTRF──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

None.

Related Information

See the following for additional information about CPP trace:

- “ZLTRL–In-Core CCP Trace Recording” on page 706 and “ZLTRN–Initiate CCP Trace to Tape” on page 708
- *TPF Non-SNA Data Communications Reference*.

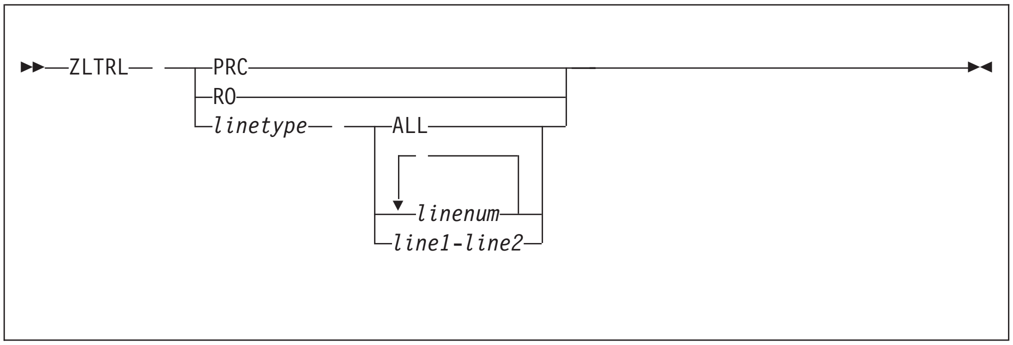
ZLTRL–In-Core CCP Trace Recording

Use this command to initiate trace recording in the *wrap around* core block mode for the specified symbolic line numbers.

Requirements and Restrictions

None.

Format



PRC
: specifies symbolic line number (SLN) 01 is to be traced. This also turns on the console trace bit for this subchannel.

RO
: specifies SLN 00 is to be traced. This also turns on the console trace bit for this subchannel.

linetype
: is the type of line to be traced (BS/AI/LC).

ALL
: indicates all valid SLNs for the specified line type are to be traced.

linenum
: is the specific line numbers to be traced; a maximum of 8 line numbers is allowed.

line1
: specifies the first line number in the range to be traced.

line2
: specifies the second line number in the range to be traced.

Additional Information

- A ZLTRL command received after a ZLTRN command will suspend recording on the RTL/RTA tape, but will continue tracing the previously specified lines in the *wrap around* core block mode.
- A ZLTRN command received after a ZLTRL command will allow tracing on the previously specified lines to continue, but will cause all trace information to be written to the RTL/RTA tape.
- The PRC/RO options are only valid when the PRC and RO are on symbolic lines 00 or 01. Where more than one ZLTRN command is given, their effect is cumulative.

Examples

None.

Related Information

See the following for additional information about CPP trace:

- “ZLTRN–Initiate CCP Trace to Tape” on page 708
- *TPF Non-SNA Data Communications Reference*.

ZLTRN–Initiate CCP Trace to Tape

Use this command to initiate CCP Trace recording to the RTL/RTA tape for the specified symbolic line numbers.

Requirements and Restrictions

None.

Format

```
►►──ZLTRN──  ──┬─PRC──────────────────────────┬─────────────────────────────►◄
               ├─RO───────────────────────────┤
               └─linetype──  ──┬─ALL──────────┤
                               │  ┌─ ───────┐ │
                               │  ▼         │ │
                               ├───linenum──┴─┤
                               └─line1-line2──┘
```

PRC
: specifies symbolic line number (SLN) 01 is to be traced. This also turns on the console trace bit for this subchannel.

RO
: specifies SLN 00 is to be traced. This also turns on the console trace bit for this subchannel.

linetype
: is the type of line to be traced (BS/AI/LC).

ALL
: indicates all valid SLNs for the specified line type are to be traced.

linenum
: is the specific line numbers to be traced; a maximum of 8 line numbers is allowed.

line1
: specifies the first line number in the range to be traced.

line2
: specifies the second line number in the range to be traced.

Additional Information

- A ZLTRL command received after a ZLTRN command will suspend recording on the RTL/RTA tape, but will continue tracing the previously specified lines in the *wrap around* core block mode.
- A ZLTRN command received after a ZLTRL command will allow tracing on the previously specified lines to continue, but will cause all trace information to be written to the RTL/RTA tape.
- The PRC/RO options are only valid when the PRC and RO are on symbolic lines 00 or 01. Where more than one ZLTRN command is given, their effect is cumulative.

Examples

None.

Related Information

See the following for additional information about CPP trace:

- “ZLTRL–In-Core CCP Trace Recording” on page 706
- *TPF Non-SNA Data Communications Reference*.

ZLTST–Perform Synchronous Link P1024 Tests

Use this command to perform SITA P1024 test functions and monitor data on synchronous link control (SLC) links.

Requirements and Restrictions

None.

Format

```
►►──ZLTST── ──┬─Ocommand─┬──────────────────────────────────────────►◄
              ├─Fcommand─┤
              ├─Dseconds─┤
              ├─Aseries──┤
              ├─Bseries──┤
              ├─W────────┤
              ├─E────────┤
              ├─OM───────┤
              └─FM───────┘
```

O performs the on command function.

F performs the off command function.

command
is a 1- to 2-digit command number.

D performs the defer command function.

seconds
is a 1- to 2-digit number that specifies the time, in seconds.

A performs the generate type A series function.

series
is a 1- to 2-digit test series number.

B performs the generate type B series function.

W performs the wait until good data message received function.

E performs the terminate wait function.

OM
starts monitoring data on the SLC links.

FM
stops monitoring data on the SLC links.

Additional Information

None.

Examples

Data is monitored on SLC links in the following example.

```
User:   ZLTST OM

System: 15.40.38 LINK MONITOR -- ON>
```

Related Information

- See the *SITA P1024 Test Guide* for more information about the SITA P1024 test functions.
- See *TPF Non-SNA Data Communications Reference* for more information about the SLC SITA P1024 test driver and monitor.

ZLVAL–Validate Communication Lines

Use this command to validate and enable binary synchronous communication (BSC) lines or ATA/IATA synchronous link control (SLC) communication lines.

Requirements and Restrictions

None.

Format

```
►►──ZLVAL── ──┬─BS── ──┬─line─┬──────────┬──────────────────────────────►◄
              │        └─ALL──┘          │
              └─AI── ──┬─line───────┬────┘
                       ├─LK── ──link─┤
                       └─ALL────────┘
```

BS
validates and enables BSC lines.

AI validates and enables ATA/IATA SLC communication lines.

line
is a 2-digit hexadecimal symbolic line number.

ALL
validates and enables all BSC lines or ATA/IATA SLC communication lines.

LK
validates all the ATA/IATA SLC communication lines on a link.

link
is a 2-digit hexadecimal symbolic link number.

Additional Information

None.

Examples

BSC line number 44 is validated and enabled in the following example.

```
User:   ZLVAL BS 44

System: 16.06.10 LVAL BS44 SUB6A BS LN VALIDATED
```

ATA/IATA SLC communication line number 35 is validated and enabled in the following example.

```
User:   ZLVAL AI 35

System: 16.08.57 LVAL AI 35    VALIDATED
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about non-SNA communications.

ZMAIL–Control the TPF Internet Mail Servers

Use this command to control the operation of the TPF Internet mail servers, which allow users, or mail clients, to send and retrieve Internet mail, also known as electronic mail (e-mail).

Requirements and Restrictions

You can enter this command only in CRAS state or higher.

Format

```
                                   ┌─ ──All────────┐
►►──ZMAIL── ──┬──┬─STARt─┬─────────┼───────────────┼──┬──────────────────────────────►◄
              │  └─STOp──┘         ├─ ──ACcept─────┤  │
              │                    ├─ ──Deliver────┤  │
              │                    └─ ──Read───────┘  │
              ├─Flush──────────────────────────────────┤
              ├─STATus─────────────────────────────────┤
              │                 ┌─OFF─┐                │
              └─SYSlog── ───────┴─ON──┴────────────────┘
```

STARt
starts the appropriate TPF Internet mail server. If you do not specify an additional parameter, all the TPF Internet mail servers are started.

STOp
stops the TPF Internet mail server after processes that are currently running end. If you do not specify an additional parameter, all the TPF Internet mail servers are stopped.

All
starts or stops the delivery of Internet mail to other servers, and starts or stops all the TPF Internet mail servers.

ACcept
starts or stops the Simple Mail Transfer Protocol (SMTP) server. If you specify this parameter with the START parameter, the SMTP server will accept and deliver new mail. If you specify this parameter with the STOP parameter, the SMTP server will no longer accept new mail. Existing mail will be delivered. If you want to stop the delivery of mail, you must specify the STOP and DELIVER parameters.

Deliver
starts or stops the delivery of Internet mail to other servers and local users. If you specify this parameter with the START parameter, the SMTP server will start delivering mail. If you specify this parameter with the STOP parameter, the SMTP server will no longer accept or deliver mail.

Read
starts or stops the Internet Message Access Protocol (IMAP) and Post Office Protocol (POP) servers. The IMAP and POP servers allow you to read Internet mail.

Flush
erases the contents of the TPF Internet mail server queue; that is, any mail that is waiting to be delivered is removed.

STATus
displays the status of the TPF Internet mail servers.

SYSlog
enables or disables syslog daemon logging for the TPF Internet mail servers, where:

OFF
disables logging for mail messages. This is the default setting when the TPF Internet mail servers are started.

ON
enables logging for mail messages.

Notes:

1. One or more TPF Internet mail servers must be running before you can specify this parameter.
2. The syslog daemon must be configured to receive mail messages. See *TPF Transmission Control Protocol/Internet Protocol* for more information about the syslog daemon.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL command is based on the postfix command of the Secure Mailer, also known as Postfix, and on the Cyrus project.
- If you must stop the Internet daemon, enter **ZMAIL STOP** to stop the TPF Internet mail servers ***before*** you enter **ZINET STOP** to stop the Internet daemon.

Examples

When you configure your TPF system for TPF Internet mail server support for the first time, you will enter a series of ZMAIL commands to do the following:

1. Initialize the database queue pointers.
2. Start the IMAP and POP servers.
3. Create accounts for your users.
4. Start all the TPF Internet mail servers.

The following example shows the sequence of these ZMAIL commands. In this example, ZMAIL STATUS is entered at various times to show the status of the TPF Internet mail servers at that time. In addition, only one mail account is created in this example.

```
User:   zmail flush

System: MAIL0001I 10.23.46 START OF RESPONSE FROM ZMAIL flush
        Active Queue has been flushed
        Deferred Queue has been flushed
        END OF DISPLAY
```

```
User:   zmail start read

System: MAIL0001I 10.41.46 START OF RESPONSE FROM ZMAIL start
        The READ MAIL service has been started
        END OF DISPLAY
        CSMP0097I 10.41.52 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0050I 10.41.52 POP3       IS NOW ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00110 PID - 40270156+
        CSMP0097I 10.41.52 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0050I 10.41.52 IMAP       IS NOW ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00143 PID - 40280011+

User:   zmail status

System: INET0031I 10.42.13 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - IMAP       PROCID - B    ACTIVATION - OPER
        PGM    - CMNA       PARM   -
        PROTOCOL   - TCP    PORT      - 00143  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582923/00C0000B COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.42.13 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0031I 10.42.13 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - POP3       PROCID - B    ACTIVATION - OPER
        PGM    - CMNC       PARM   -
        PROTOCOL   - TCP    PORT      - 00110  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582922/00C0000A COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.42.13 CPU-B SS-BSS  SSU-HPN  IS-01
        MAIL0001I 10.42.13 START OF RESPONSE FROM ZMAIL status
        The ACCEPT MAIL service is not running
        The DELIVER MAIL service is not running
        The READ MAIL service has been started
        END OF DISPLAY
```

```
User:   zmail cm steve

System: MAIL0003I 15:00:31 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail password steve stevepwd

System: MAIL0003I 15:00:31 password COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

```
User:   zmail start

System: MAIL0001I 10.43.46 START OF RESPONSE FROM ZMAIL start
        The ACCEPT MAIL service has been started
        The DELIVER MAIL service has been started
        END OF DISPLAY
        MAIL0002E 10.43.46 START OF ERROR RESPONSE FROM ZMAIL start
        The READ MAIL service has already been started
        END OF DISPLAY+
        CSMP0097I 10.43.52 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0050I 10.43.52 SMTP       IS NOW ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00025 PID - 402900F0+
User:   zmail status

System: INET0031I 10.26.35 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - SMTP       PROCID - B    ACTIVATION - OPER
        PGM    - CMN0       PARM   -
        PROTOCOL   - TCP    PORT      - 00025  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582918/00C00006 COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.26.35 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0031I 10.26.35 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - IMAP       PROCID - B    ACTIVATION - OPER
        PGM    - CMNA       PARM   -
        PROTOCOL   - TCP    PORT      - 00143  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582917/00C00005 COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.27.01 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0031I 10.27.01 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - POP3       PROCID - B    ACTIVATION - OPER
        PGM    - CMNC       PARM   -
        PROTOCOL   - TCP    PORT      - 00110  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582919/00C00007 COUNT - 0000000000
                                                  TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.27.01 CPU-B SS-BSS  SSU-HPN  IS-01
        MAIL0001I 10.27.01 START OF RESPONSE FROM ZMAIL status
        The ACCEPT MAIL service has been started
        The DELIVER MAIL service has been started
        The READ MAIL service has been started
        END OF DISPLAY
```

In the following example, all the TPF Internet mail servers are stopped.

```
User:   zmail stop

System: MAIL0001I 10.27.54 START OF RESPONSE FROM ZMAIL stop
        The ACCEPT MAIL service has been stopped
        The DELIVER MAIL service has been stopped
        The READ MAIL service has been stopped
        END OF DISPLAY
        CSMP0097I 10.27.54 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0051I 10.27.54 IMAP       IS NO LONGER ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00143 PID - 4019010A+
        CSMP0097I 10.27.54 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0051I 10.27.54 SMTP       IS NO LONGER ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00025 PID - 40190148+
        CSMP0097I 10.27.54 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0051I 10.27.54 POP3       IS NO LONGER ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00110 PID - 40190149+
```

In the following example, the SMTP server is started and will begin to accept and deliver mail. In this example, the IMAP and POP servers are not running.

```
User:   zmail start accept

System: MAIL0001I 10.31.10 START OF RESPONSE FROM ZMAIL start
        The ACCEPT MAIL service has been started
        The DELIVER MAIL service has been started
        END OF DISPLAY
        CSMP0097I 10.31.16 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0050I 10.31.16 SMTP       IS NOW ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00025 PID - 401F01E7+

User:   zmail status

System: INET0031I 10.39.47 START OF ZINET DISPLAY OF ACTIVE SERVER
        SERVER - SMTP       PROCID - B    ACTIVATION - OPER
        PGM    - CMN0       PARM   -
        PROTOCOL   - TCP    PORT      - 00025  MODEL - NOWAIT   _
        SERVERRORS - 00000  SERVETIME - 00000  USER  - nobody
        MAXPROC    - 00000  TIMEOUT   - 00000  STATE - NORM
        AORLENGTH  - 00000
        IP - ANY             SOCKET - 12582921/00C00009 COUNT - 0000000000
                                               TOTAL COUNT - 0000000000
        END OF DISPLAY
        CSMP0097I 10.39.47 CPU-B SS-BSS  SSU-HPN  IS-01
        MAIL0001I 10.39.47 START OF RESPONSE FROM ZMAIL status
        The ACCEPT MAIL service has been started
        The DELIVER MAIL service has been started
        The READ MAIL service is not running
        END OF DISPLAY
```

In the following example, the accept mail service is stopped; that is, the SMTP server will no longer accept new mail. Existing mail will continue to be delivered.

```
User:   zmail stop accept

System: MAIL0001I 10.40.23 START OF RESPONSE FROM ZMAIL stop
        The ACCEPT MAIL service has been stopped
        END OF DISPLAY
        CSMP0097I 10.40.24 CPU-B SS-BSS  SSU-HPN  IS-01
        INET0051I 10.40.24 SMTP       IS NO LONGER ACCEPTING CONNECTIONS ON
                           IP - ANY PORT - 00025 PID - 402500BD+

User:   zmail status

System: MAIL0001I 10.40.31 START OF RESPONSE FROM ZMAIL status
        The ACCEPT MAIL service is not running
        The DELIVER MAIL service has been started
        The READ MAIL service is not running
        END OF DISPLAY
```

In the following example, delivery of mail is stopped entirely; that is, the SMTP server will no longer accept or deliver mail.

```
User:   zmail stop deliver

System: MAIL0001I 10.41.07 START OF RESPONSE FROM ZMAIL stop
        The DELIVER MAIL service has been stopped
        END OF DISPLAY
```

In the following example, logging to the syslog daemon for the TPF Internet mail servers is started. Any mail messages that are generated by the TPF Internet mail servers after this command is entered will be logged to the syslog daemon.

```
User:   zmail sys on

System: MAIL0003I 09.11.03 sys COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support, including how to configure your TPF system to use the servers.
- See http://www.postfix.org for more information about the Secure Mailer.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL CREATEMAILBOX–Create a Mailbox” on page 720 for more information about creating accounts for your users.

ZMAIL CREATEMAILBOX–Create a Mailbox

Use this command to create a new mailbox in a TPF Internet mail server database.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain in which you want to create the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL CREATEMAILBOX─┬─ ──mailbox──────────────────────────────────────►◄
    └─ZMAIL CM────────────┘
```

Note: You can use ZMAIL CM as a short form of the ZMAIL CREATEMAILBOX command.

mailbox
: is the name of the mailbox that you want to create. If this is the first mailbox for a user, specify the account name for that user. If this is a mailbox for an existing user account (referred to as a *submailbox*), specify the mailbox name in the form *account***.inbox.***submailbox*.

  For example, to create the first mailbox for user1, enter:

  zmail cm user1

  To create a submailbox for user1, enter:

  zmail cm user1.inbox.another

  You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL CREATEMAILBOX command is based on the createmailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, a new mailbox for an account named `kate` is created. A ZMAIL LISTMAILBOX command (ZMAIL LM) is entered to verify that the mailbox was created.

```
User:   zmail cm kate

System: MAIL0003I 15:00:31 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail lm kate

System: MAIL0001I 15:01:35 START OF RESPONSE FROM ZMAIL lm
        kate.inbox
        END OF DISPLAY
```

In the following example, submailboxes are created for the account named `kate`. A ZMAIL LISTMAILBOX command (ZMAIL LM) is entered to verify that the submailboxes were created.

```
User:   zmail cm kate.inbox.work

System: MAIL0003I 15:00:31 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail cm kate.inbox.work.project1

System: MAIL0003I 15:01:00 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail cm kate.inbox.play

System: MAIL0003I 15:01:30 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail lm kate

System: MAIL0001I 15:01:55 START OF RESPONSE FROM ZMAIL lm
        kate.inbox
        kate.inbox.work
        kate.inbox.work.project1
        kate.inbox.play
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL LISTMAILBOX–Display a List of Mailboxes” on page 728 for more information about listing the mailboxes for an account.

ZMAIL DELETEACLMAILBOX–Delete a User from a Mailbox

Use this command to delete a user from the access control list (ACL) for a mailbox in a TPF Internet mail server database. An *access control list* specifies the users who have permission to access the mailboxes. The access for a mailbox is set by using the ZMAIL SETACLMAILBOX command.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
                                                 ┌──────────┐
                                                 ▼          │
►►──┬─ZMAIL DELETEACLMAILBOX─┬── ──mailbox─── ──account─┴───────────────►◄
    └─ZMAIL DAM──────────────┘
```

Note: You can use ZMAIL DAM as a short form of the ZMAIL DELETEACLMAILBOX command.

mailbox
: is the name of the mailbox for which you want to change the access.

account
: is the account name for the user that you want to delete from the access control list.

You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL DELETEACLMAILBOX command is based on the deleteaclmailbox command of the Cyrus project.
- You can only delete access rights for users in your own domain.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, the access rights for mailbox `kate.inbox.work` for account `steve` are deleted.

```
User:   zmail deleteaclmailbox kate.inbox.work steve

System: MAIL0003I 13:11:31 deleteaclmailbox COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL SETACLMAILBOX–Set the List of Users for a Mailbox” on page 742 for more information about setting the access for a mailbox.

ZMAIL DELETEMAILBOX–Delete a Mailbox

Use this command to delete a mailbox from a TPF Internet mail server database.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL DELETEMAILBOX─┬── ──mailbox───────────────────────────────►◄
    └─ZMAIL DM────────────┘
```

Note: You can use ZMAIL DM as a short form of the ZMAIL DELETEMAILBOX command.

mailbox
is the name of the mailbox that you want to delete. If you specify an account name for the mailbox, the account and all the mailboxes associated with that account are deleted.

You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL DELETEMAILBOX command is based on the deletemailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, a submailbox named `kate.inbox.work.project1` is deleted.

```
User:   zmail lm kate

System: MAIL0001I 15:01:35 START OF RESPONSE FROM ZMAIL lm
        kate.inbox
        kate.inbox.work
        kate.inbox.work.project1
        kate.inbox.work.project1.final
        kate.inbox.play
        END OF DISPLAY

User:   zmail dm kate.inbox.work.project1

System: MAIL0003I 15:00:31 dm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail lm kate

System: MAIL0001I 15:01:35 START OF RESPONSE FROM ZMAIL lm
        kate.inbox
        kate.inbox.work
        kate.inbox.work.project1.final
        kate.inbox.play
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL CREATEMAILBOX–Create a Mailbox” on page 720 for more information about creating mailboxes.
- See “ZMAIL LISTMAILBOX–Display a List of Mailboxes” on page 728 for more information about listing the mailboxes for an account.

ZMAIL LISTACLMAILBOX–Display Users for a Mailbox

Use this command to display a list of users with access to a mailbox in a TPF Internet mail server database. The access to a mailbox is controlled by an access control list (ACL). You can set or change the ACL by using the ZMAIL SETACLMAILBOX command.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL LISTACLMAILBOX─┬── ──mailbox──────────────────────────────►◄
    └─ZMAIL LAM────────────┘
```

Note: You can use ZMAIL LAM as a short form of the ZMAIL LISTACLMAILBOX command.

mailbox
is the name of the mailbox for which you want a list of users.

You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL LISTACLMAILBOX command is based on the listaclmailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, a list of users with access to mailbox `tpfuser` is displayed. Each line of the display shows the account name of the user followed by the access rights of that user for the specified mailbox. See “ZMAIL SETACLMAILBOX–Set the List of Users for a Mailbox” on page 742 for an explanation of the actual access rights shown in the example.

```
User:   ZMAIL listaclmailbox tpfuser

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL listaclmailbox
        tpfuser lrswipcda
        rocky lr
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See "ZMAIL SETACLMAILBOX–Set the List of Users for a Mailbox" on page 742 for more information about setting the access for a mailbox.

ZMAIL LISTMAILBOX–Display a List of Mailboxes

Use this command to display a list of mailboxes for an account in a TPF Internet mail server database.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL LISTMAILBOX─┬── ──mailbox──────────────────────────────────►◄
    └─ZMAIL LM──────────┘
```

Note: You can use ZMAIL LM as a short form of the ZMAIL LISTMAILBOX command.

mailbox
is the mailbox that you want to display. You can use the following wildcard characters in the mailbox name to display mailboxes that match a particular name pattern:

Wildcard	Description
mailbox.*	Matches any string.
mailbox.%	Matches any string except the hierarchy separator. For example, if you enter **zmail listmailbox user.inbox.foo.%** and mailboxes `user.inbox.foo.bar` and `user.inbox.foo.bar.old` both exist, `user.inbox.foo.bar` is then displayed, but `user.inbox.foo.bar.old` is not. Sometimes, when the % character is used as the last character of a pattern, nonmailbox names are listed in parentheses. This indicates that the name is not actually a mailbox, but there are submailboxes under that name. For example, if you enter **zmail listmailbox user.inbox.foo.%** and mailbox `user.inbox.foo.bar.old` exists, but `user.inbox.foo.bar` does not exist, (`user.inbox.foo.bar`) is then displayed.

If you use a wildcard character, you must specify at least the account name.

You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

ZMAIL HELP
ZMAIL ?

- The ZMAIL LISTMAILBOX command is based on the listmailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, all the submailboxes for mailbox `curly` are displayed.

```
User:   zmail lm curly

System: MAIL0001I 10:01:35 START OF RESPONSE FROM ZMAIL lm
        curly.inbox
        curly.inbox.foo
        curly.inbox.foo.bar
        curly.inbox.foo.blah
        curly.inbox.foo.bar.old
        END OF DISPLAY
```

In the following example, all the submailboxes that begin with the string `curly.inbox.foo.b` are displayed.

```
User:   zmail lm curly.inbox.foo.b*

System: MAIL0001I 10:01:35 START OF RESPONSE FROM ZMAIL lm
        curly.inbox.foo.bar
        curly.inbox.foo.blah
        curly.inbox.foo.bar.old
        END OF DISPLAY
```

In the following example, all the submailboxes that begin with the string `curly.inbox.foo.b` at the same level are displayed.

```
User:   zmail lm curly.inbox.foo.b%

System: MAIL0001I 10:01:35 START OF RESPONSE FROM ZMAIL lm
        curly.inbox.foo.bar
        curly.inbox.foo.blah
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.

ZMAIL LISTQUOTA–Display Storage for a Mailbox

Use this command to display the amount of storage currently being used by, and the maximum amount of storage allowed for, one or more mailboxes in a TPF Internet mail server database. The maximum amount of storage allowed for mailboxes is referred to as the *quota*. A quota is applied to a *quota root*, which is the point in a mailbox naming hierarchy where the quota (storage limit) is set. A quota root is set when you enter the ZMAIL SETQUOTA command. See “ZMAIL SETQUOTA–Set the Storage Limit for a Mailbox” on page 745 for more information about quotas and quota roots.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.
- You can enter this command only for a mailbox name or placeholder name for which a quota was set with the ZMAIL SETQUOTA command. If you want to display the storage for a mailbox in the hierarchy for which there is no specific quota set, use the ZMAIL LISTQUOTAROOT command.

Format

```
►►──┬─ZMAIL LISTQUOTA─┬── ──mailbox─────────────────────────────────────►◄
    └─ZMAIL LQ────────┘
```

Note: You can use ZMAIL LQ as a short form of the ZMAIL LISTQUOTA command.

mailbox
: is the name of the mailbox or the name of the placeholder in the mailbox hierarchy for which you want to display storage. The amount of storage currently being used for mail items in, and the maximum amount of storage allowed for, the specified mailbox hierarchy is displayed. Only mailboxes (or placeholder names) that have a set quota (or storage limit) are displayed. See the ZMAIL SETQUOTA command for more information about setting a quota for mailboxes.

  You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL LISTQUOTA command is based on the listquota command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

The storage for a mailbox is shown in the following format:

used/*max*

where:

used
: Shows the number of 4-KB pool records currently being used to hold mail items in the specified mailbox hierarchy.

max
: Shows the maximum number of 4-KB pool records allowed for the specified mailbox hierarchy.

In the following example, assume the following mailboxes exist:

```
tpfuser.inbox
tpfuser.inbox.work.project1
tpfuser.inbox.work.project2
tpfuser.inbox.play
tpfuser.inbox.todo
```

Also, assume that quotas were previously set as follows:
- `tpfuser.inbox` has a quota of 1000
- `tpfuser.inbox.work` has a quota of 5000
- `tpfuser.inbox.play` has a quota of 3000

Therefore, in this example, mailboxes `tpfuser.inbox` and `tpfuser.inbox.todo` can use a maximum of 1000 4-KB pool records to hold mail items. Mailboxes `tpfuser.inbox.work.project1` and `tpfuser.inbox.work.project2` can use a maximum of 5000 4-KB pool records, and mailbox `tpfuser.inbox.play` can use a maximum of 3000 4-KB pool records. The following displays show these storage limits and the amount of storage currently being used by these mailboxes. This example also shows the error message that is displayed if you enter the ZMAIL LISTQUOTA command for a mailbox in the hierarchy for which there is no specific quota set.

```
User:   zmail listquota tpfuser

System: MAIL0001I 13:11:30 START OF RESPONSE FROM ZMAIL listquota
        STORAGE 400/1000
        END OF DISPLAY

User:   zmail lq tpfuser.inbox.work

System: MAIL0001I 13:11:45 START OF RESPONSE FROM ZMAIL lq
        STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lq tpfuser.inbox.work.project1

System: MAIL0002E 13:12:15 START OF ERROR RESPONSE FROM ZMAIL lq
        command failed: Quota root does not exist
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.work.project1

System: MAIL0001I 13:12:16 START OF RESPONSE FROM ZMAIL lqr
        inbox.work STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.work.project2

System: MAIL0001I 13:12:45 START OF RESPONSE FROM ZMAIL lqr
        inbox.work STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lq tpfuser.inbox.play

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL lq
        STORAGE 500/3000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.todo

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL lqr
        inbox STORAGE 200/1000
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL LISTQUOTAROOT–Display the Quota Root for a Mailbox” on page 733 for more information about displaying storage for mailboxes.
- See “ZMAIL SETQUOTA–Set the Storage Limit for a Mailbox” on page 745 for more information about setting storage limits for mailboxes.

ZMAIL LISTQUOTAROOT–Display the Quota Root for a Mailbox

Use this command to display the quota root for a mailbox, as well as the amount of storage currently being used by, and the maximum amount of storage allowed for, the mailboxes associated with that quota root. The maximum amount of storage allowed for one or more mailboxes is referred to as the *quota*. A quota is applied to a *quota root*, which is the point in a mailbox naming hierarchy where the quota (storage limit) is set. A quota root is set when you enter the ZMAIL SETQUOTA command. See “ZMAIL SETQUOTA–Set the Storage Limit for a Mailbox” on page 745 for more information about quotas and quota roots.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.
- You can enter this command only for an actual mailbox. If you want to display the quota for a placeholder in the mailbox naming hierarchy, use the ZMAIL LISTQUOTA command.

Format

```
►►──┬─ZMAIL LISTQUOTAROOT─┬── ──mailbox──────────────────────────────►◄
    ├─ZMAIL LQR───────────┤
    └─ZMAIL LQM───────────┘
```

Note: You can use ZMAIL LQR or ZMAIL LQM as a short form of the ZMAIL LISTQUOTAROOT command.

mailbox
: is the name of the mailbox for which you want to display the quota root. In addition to the quota root, the amount of storage currently being used by, and the maximum amount of storage allowed for, the mailboxes associated with that quota root is displayed.

  You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL LISTQUOTAROOT command is based on the listquotaroot command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

The storage for a mailbox is shown in the following format:

used/*max*

where:

used
: Shows the number of 4-KB pool records currently being used to hold mail items in the mailboxes associated with the quota root of the specified mailbox.

max
: Shows the maximum number of 4-KB pool records allowed for the mailboxes associated with the quota root of the specified mailbox.

In the following examples, assume the following mailboxes exist:

```
tpfuser.inbox
tpfuser.inbox.work.project1
tpfuser.inbox.work.project2
tpfuser.inbox.play
tpfuser.inbox.todo
```

Also, assume that quotas were previously set as follows:
- `tpfuser.inbox` has a quota of 1000
- `tpfuser.inbox.work` has a quota of 5000
- `tpfuser.inbox.play` has a quota of 3000

Therefore, in this example, mailboxes `tpfuser.inbox` and `tpfuser.inbox.todo` can use a maximum of 1000 4-KB pool records to hold mail items. Mailboxes `tpfuser.inbox.work.project1` and `tpfuser.inbox.work.project2` can use a maximum of 5000 4-KB pool records, and mailbox `tpfuser.inbox.play` can use a maximum of 3000 4-KB pool records. The following displays show the quota roots (the point where the storage limit is set) for each of the mailboxes, the amount of storage currently being used by these mailboxes, and the maximum amount of storage allowed for the mailboxes. This example also shows the error message that is displayed if you enter the ZMAIL LISTQUOTAROOT command for a placeholder name instead of an actual mailbox name.

```
User:   zmail listquotaroot tpfuser

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL listquotaroot
        inbox STORAGE 400/1000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.work

System: MAIL0002E 13:11:55 START OF ERROR RESPONSE FROM ZMAIL lqr
        command failed: Mailbox does not exist
        END OF DISPLAY

User:   zmail lq tpfuser.inbox.work

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL lq
        STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.work.project1

System: MAIL0001I 13:12:16 START OF RESPONSE FROM ZMAIL lqr
        inbox.work STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.work.project2

System: MAIL0001I 13:12:45 START OF RESPONSE FROM ZMAIL lqr
        inbox.work STORAGE 3500/5000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.play

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL lqr
        inbox.play STORAGE 500/3000
        END OF DISPLAY

User:   zmail lqr tpfuser.inbox.todo

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL lqr
        inbox STORAGE 200/1000
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL LISTQUOTA–Display Storage for a Mailbox” on page 730 for more information about displaying storage for mailboxes.
- See “ZMAIL SETQUOTA–Set the Storage Limit for a Mailbox” on page 745 for more information about setting storage limits for mailboxes.

ZMAIL PASSWORD–Set or Change the Password for an Account

Use this command to set or change the password for an account in a TPF Internet mail server database.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL PASSWORD─┬── ──account── ──password──────────────────────────►◄
    └─ZMAIL PW───────┘
```

Note: You can use ZMAIL PW as a short form of the ZMAIL PASSWORD command.

account
is the name of the account for which you want to set or change the password.

You can specify as many as 30 characters for an account name. The name of an account can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

password
is a 1- to 8-character password for the account. The password can consist of any character allowed by the computer room agent set (CRAS) terminal.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, a new account named `teddybear` is created and the password is set to cuddles.

```
User:   zmail cm teddybear

System: MAIL0003I 15:00:31 cm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail password teddybear cuddles

System: MAIL0003I 15:00:31 password COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.

- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.

ZMAIL PATH–Display the Path to an Account

Use this command to examine internal mail structures or file record contents in a TPF Internet mail server database. This command displays the path through the first-level and second-level index to get to the user profile record (UPR) for a specified account.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the account. Use the ZFILE export command to set the environment variable.

Format

```
►►──ZMAIL PATH── ──account──────────────────────────────────────────────►◄
```

account
is the name of the account for which you want to display the path.

You can specify as many as 30 characters for an account name. The name of an account can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

The ZPATH command display provides the following information.

Note: Not all of the following fields appear in every display.

HASH VALUE
is a number calculated for the specified account. This number is used to locate the UPR.

LEVEL1 RCD
is the ordinal number of the #MAILxx fixed file record; that is, the first-level index to the UPR.

LEVEL1 ENTRY
is the slot number in the fixed file record that contains the file address of either the second-level index to the UPR or the UPR account record.

LEVEL2 RCD
is the quotient when the hash value is divided for the first time. This quotient is then divided by the number of slots in the first-level index records to determine LEVEL1 RCD and LEVEL1 ENTRY.

LEVEL2 ENTRY
: is the slot number in the second-level index to the UPR if the first-level index slot contains a UPR second-level index file address.

LEVEL1 INDEX FA
: is the file address of the first-level index record for the UPR.

LEVEL2 INDEX FA
: is the file address of the secound-level index record for the UPR.

CONTENTION RECORD FA
: is the file address of the contention table record for the UPR. A contention table occurs when 2 user names hash to the same level 2 slot when that level 2 slot is in use.

USER PROFILE RECORD FA
: is the file address of the UPR for this account.

END OF PATH, FOUND
: indicates that there is a UPR for the specified account.

END OF PATH, NOT FOUND
: indicates that there is no UPR for the specified account, but shows the path if that account did exist.

In the following example, the path for account `user1000` is displayed:

```
User:   zmail path user1000

System: MAIL0100I 12.46.40 PATH FOR UID user1000
        HASH VALUE 96937689
        LEVEL1 RCD 00000068 LEVEL1 ENTRY 000001ED
        LEVEL2 RCD 0000CD0D LEVEL2 ENTRY 0000007F
        LEVEL1 INDEX FA F40D99A1
        LEVEL2 INDEX FA 001653A5
        USER PROFILE RECORD FA 08275241
        END OF PATH, FOUND
        END OF DISPLAY
```

In the following example, the path for account `aaaabbbbcccc` is displayed:

```
User:   zmail path aaaabbbbcccc

System: MAIL0100I 12.48.08 PATH FOR UID aaaabbbbcccc
        HASH VALUE 87878786
        LEVEL1 RCD 0000005F LEVEL1 ENTRY 00000070
        LEVEL2 RCD 0000B9FC LEVEL2 ENTRY 00000127
        LEVEL1 INDEX FA F40D997D
        LEVEL2 INDEX FA 00165381
        CONTENTION RECORD FA 0827522D
        USER PROFILE RECORD FA 082751C5
        END OF PATH, FOUND
        END OF DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.

ZMAIL RENAMEMAILBOX–Change the Name of a Submailbox

Use this command to change the name of a submailbox in a TPF Internet mail server database.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.
- You cannot rename an account. You can only rename submailboxes for an account.

Format

```
►►──┬─ZMAIL RENAMEMAILBOX─┬── ──submailbox── ──newsubmailbox─────────────────►◄
    └─ZMAIL RENM──────────┘
```

Note: You can use ZMAIL RENM as a short form of the ZMAIL RENAMEMAILBOX command.

submailbox
is the name of the submailbox whose name you want to change.

newsubmailbox
is the new submailbox name.

The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of a mailbox can consist of any alphanumeric character, an underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL RENAMEMAILBOX command is based on the renamemailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, submailbox `steve.inbox.work` is renamed to `steve.inbox.workstuff`.

```
User:   zmail lm steve

System: MAIL0001I 14:00:45 START OF RESPONSE FROM ZMAIL lm
        steve.inbox
        steve.inbox.work
        steve.inbox.play
        END OF DISPLAY

User:   zmail renm steve.inbox.work steve.inbox.workstuff

System: MAIL0003I 14:01:35 renm COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail lm steve

System: MAIL0001I 14:02:00 START OF RESPONSE FROM ZMAIL lm
        steve.inbox
        steve.inbox.workstuff
        steve.inbox.play
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL CREATEMAILBOX–Create a Mailbox” on page 720 for more information about creating mailboxes.
- See “ZMAIL LISTMAILBOX–Display a List of Mailboxes” on page 728 for more information about listing the mailboxes for an account.

ZMAIL SETACLMAILBOX–Set the List of Users for a Mailbox

Use this command to set or change the access control list (ACL) for a mailbox in a TPF Internet mail server database. An *access control list* specifies the users who have permission to access the mailboxes.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
                                               ┌─ ────────────────────────┐
                                               ▼                          │
►►──┬─ZMAIL SETACLMAILBOX─┬── ──mailbox── ──────account── ──rights──┴──────────────►◄
    └─ZMAIL SAM───────────┘
```

Note: You can use ZMAIL SAM as a short form of the ZMAIL SETACLMAILBOX command.

mailbox
: is the mailbox for which you want to set or change the ACL entry.

account
: is the account name of the user for which the ACL entry applies.

You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

rights
: is the access right for the specified user. Specify the access right as one or more of the following letters:

l specifies *lookup* access, which allows you to see that the mailbox exists.

r specifies *read* access, which allows you to read the mailbox. You can select the mailbox, retrieve data, perform searches, and copy messages from the mailbox.

s specifies *seen* access, which allows you to check if the mail has been read (seen) or not read (not seen).

w specifies *write* access, which allows you to modify the following properties associated with mail items in the specified mailbox:

answered
: Indicates if the mail has been answered.

flagged
: Indicates that a mail item is marked urgent or requires special attention.

draft Indicates that this is a draft mail item; that is, the composition of the message is not complete.

These flags are modified through the Internet Message Access Protocol (IMAP) client.

i specifies *insert* access, which allows you to add (insert) new mail items to the mailbox.

p specifies *post* access.

Note: This access right is not meaningful for TPF mail.

c specifies *create* access, which allows you to create new submailboxes of the mailbox.

d specifies *delete* access, which allows you to delete mail items.

a specifies *administer* access, which allows you to change the access control list for the mailbox.

You can also specify the access rights by using the following value:

all specifies all (**lrswipcda**) access rights.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL SETACLMAILBOX command is based on the setaclmailbox command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.

Examples

In the following example, account `rocky` is given look and read access to the `tpfuser` mailbox. A ZMAIL LISTACLMAILBOX command is entered to verify the updated access rights.

```
User:   zmail sam tpfuser rocky lr

System: MAIL0003I 13:11:31 sam COMPLETED SUCCESSFULLY. NO OUTPUT TO DISPLAY

User:   zmail listaclmailbox tpfuser

System: MAIL0001I 13:11:31 START OF RESPONSE FROM ZMAIL listaclmailbox
        tpfuser lrswipcda
        rocky lr
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See “ZMAIL DELETEACLMAILBOX–Delete a User from a Mailbox” on page 722 for more information about deleting a user from the ACL.

- See “ZMAIL LISTACLMAILBOX–Display Users for a Mailbox” on page 726 for more information about listing the users that have access to a mailbox.

ZMAIL SETQUOTA–Set the Storage Limit for a Mailbox

Use this command to set the quota (storage limit) for one or more mailboxes in a TPF Internet mail server database. The *quota* is the maximum amount of storage allowed for one or more mailboxes. A quota is applied to a *quota root*, which is the point in a mailbox naming hierarchy where the quota (storage limit) is set. A quota root does not have to be a mailbox.

Quotas on a quota root apply to the total amount of storage used by any mailbox at that level and any submailboxes of that level that are not under a quota root in a lower-level hierarchy. For example, assume you have the following mailboxes in your mail database:

```
tpfuser.inbox
tpfuser.inbox.work.project1
tpfuser.inbox.work.project2
tpfuser.inbox.play
tpfuser.inbox.todo
```

Also assume that a quota is set at the following points in the naming hierarchy:

```
tpfuser.inbox
tpfuser.inbox.work
tpfuser.inbox.play
```

These points are referred to as quota roots. In this example, the storage limit set for quota root `tpfuser.inbox` applies to the mailboxes named `tpfuser.inbox` and `tpfuser.inbox.todo`; the storage limit set for quota root `tpfuser.inbox.work` applies to the mailboxes named `tpfuser.inbox.work.project1` and `tpfuser.inbox.work.project2`; and the storage limit set for quota root `tpfuser.inbox.play` applies to the mailbox named `tpfuser.inbox.play`.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You must set the MAILDOMAIN environment variable to the domain that contains the mailbox. Use the ZFILE export command to set the environment variable.

Format

```
►►──┬─ZMAIL SETQUOTA─┬── ──mailbox── ──quota──────────────────────────►◄
    └─ZMAIL SQ───────┘
```

Note: You can use ZMAIL SQ as a short form of the ZMAIL SETQUOTA command.

mailbox
: is the point in the mailbox naming hierarchy for which you want to set the storage limit, or quota.

  You can specify as many as 30 characters for an account name. The total length of a mailbox string (including all submailboxes) can be as many as 256 characters. The name of an account or mailbox can consist of any alphanumeric character, underscore (_), hyphen (-), comma (,), semicolon (;), left parenthesis (( ), or right parenthesis ( )).

quota
is the storage limit for this resource. Specify the maximum number of 4-KB pool records that the mailbox (or mailboxes in the hierarchy) can use. A quota value of 0 indicates that there is no storage limit.

If you specify a value that is below the amount of storage currently being used, the next mail item will be rejected. Subsequent mail will be rejected until the amount of storage being used falls below the new quota.

The first mail item that causes the quota to be exceeded will be accepted and any mail following that will be rejected.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMAIL HELP
  ZMAIL ?
- The ZMAIL SETQUOTA command is based on the setquota command of the Cyrus project.
- All data entered with the ZMAIL commands is changed to lowercase.
- Use the ZMAIL LISTQUOTA and ZMAIL LISTQUOTAROOT commands to display the amount of storage being used and the maximum amount of storage allowed for the mailboxes.

Examples

In the following example, the storage limits are set for various points in the mailbox naming hierarchy. See “ZMAIL LISTQUOTA–Display Storage for a Mailbox” on page 730 and “ZMAIL LISTQUOTAROOT–Display the Quota Root for a Mailbox” on page 733 for examples of displaying these storage limits and the amount of storage being used to hold mail items in the mailboxes.

```
User:   zmail lm tpfuser

System: MAIL0001I 10:05:30 START OF RESPONSE FROM ZMAIL lm
        tpfuser.inbox
        tpfuser.inbox.work.project1
        tpfuser.inbox.work.project2
        tpfuser.inbox.play
        tpfuser.inbox.todo
        END OF DISPLAY

User:   zmail setquota tpfuser 1000

System: MAIL0001I 10:05:49 START OF RESPONSE FROM ZMAIL setquota
        STORAGE 0/1000
        END OF DISPLAY

User:  zmail sq tpfuser.inbox.work 5000

System: MAIL0001I 10:06:20 START OF RESPONSE FROM ZMAIL sq
        STORAGE 0/5000
        END OF DISPLAY

User:  zmail sq tpfuser.inbox.play 3000

System: MAIL0001I 10:06:20 START OF RESPONSE FROM ZMAIL sq
        STORAGE 0/3000
        END OF DISPLAY
```

Related Information

- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TPF Internet mail server support.
- See http://www.cmu.edu/computing/cyrus for more information about the Cyrus project.
- See "ZMAIL LISTQUOTA–Display Storage for a Mailbox" on page 730 and "ZMAIL LISTQUOTAROOT–Display the Quota Root for a Mailbox" on page 733 for more information about displaying storage limits for mailboxes.

ZMATP–Display or Modify MATIP Values

Use this command to do the following:

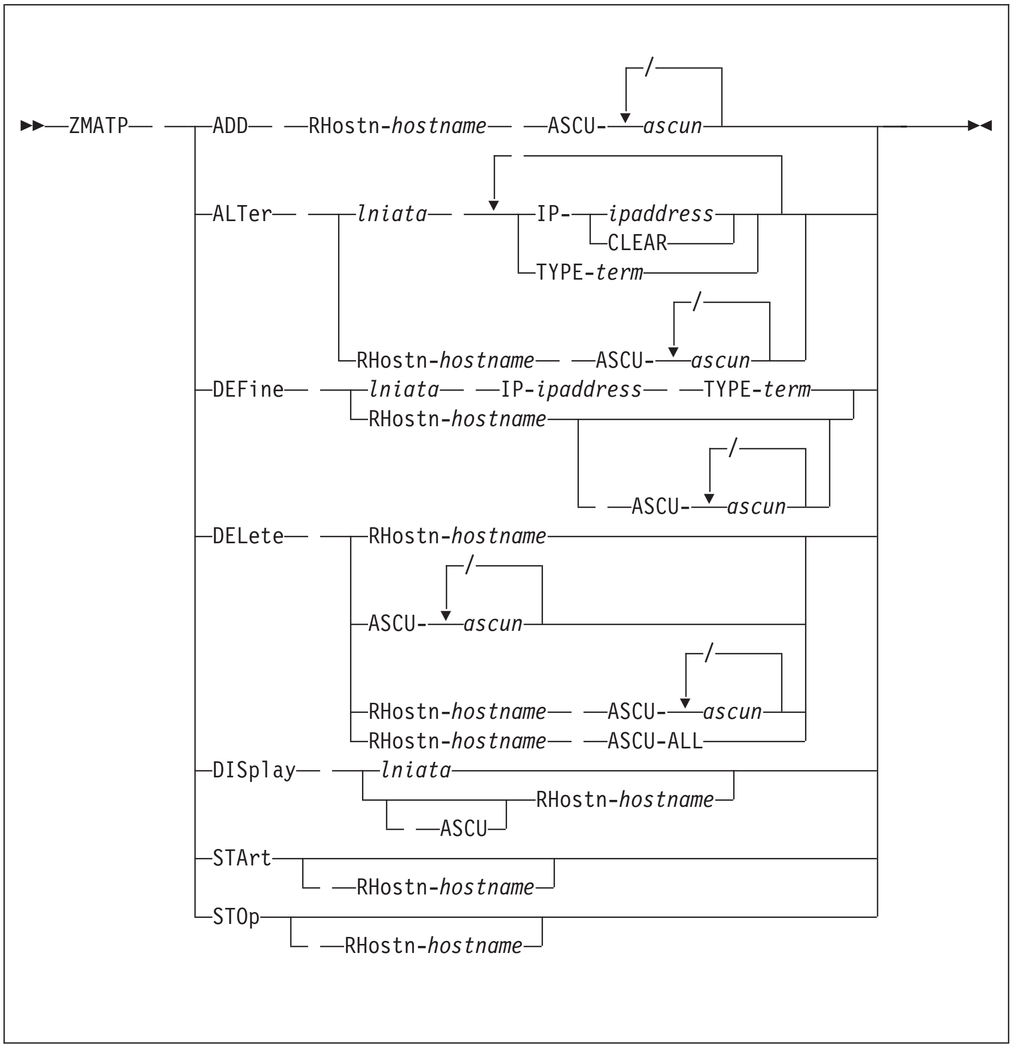
- Add host names and agent set control units (ASCUs).
- Change or define an Internet Protocol (IP) address or a terminal type for a specified line number, interchange address, and terminal address (LNIATA). This enables messages destined for the terminal to be routed by the specified Transmission Control Protocol/Internet Protocol (TCP/IP) middleware (MATIP Type A, MATIP Type B, or IP Bridge).
- Change or define host names or ASCUs.
- Delete a host name or ASCU.
- Display session information for active terminals that support Mapping of Airline Traffic over Internet Protocol (MATIP) by using the LNIATA or the host name.
- Start the MATIP Type-A and MATIP Type-B listeners and activate MATIP support.
- Update MATIP session characteristics in core.
- Stop the MATIP Type-A and MATIP Type-B listeners and close all active MATIP sessions.
- Stop MATIP sessions by using the host name.
- Define MATIP printer devices.

MATIP provides a means for using TCP/IP communications for legacy communication protocols such as X.25.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- MATIP sessions between the same host name are not allowed.
- Host names are not defined unless they can be resolved by the DNS name server defined in your TPF 4.1 system.
- If you add host names when MATIP is running, you must stop and then start MATIP again before you can use the host names.
- The terminal type in the WGTA table entry is not updated unless a hardcoded (permanent) terminal type is entered, or if the previous terminal type was associated with a TCP/IP middleware device.

Format



ADD
: adds ASCUs to a host name. The host name must be already defined.

RHostn-*hostname*
: specifies the remote host name, where *hostname* is the name of the host. Specify the host name in alphanumeric characters. Each portion of the host name separated by dots must be no larger than 63 alphanumeric characters.

  Note: Hyphens (-) are also allowed in the host name.

ASCU-
: specifies the ASCU, where:

  ascun
  : specifies the list of ASCUs. The maximum number of ASCUs in a specified list is eleven 2-byte ASCUs or six 4-byte ASCUs.

  ALL
  : deletes every ASCU associated with a specific host name.

ALTer
changes the IP address, terminal type, or both, associated with the LNIATA specified or changes the existing host name. The host name must be already defined.

lniata
is a 6-digit hexadecimal number for the LNIATA.

IP-
specifies if the IP address is to be added to the terminal address table (WGTA) entry for the LNIATA specified, where:

ipaddress
is an IP address in dotted decimal notation.

CLEAR
specifies that no IP address should be associated with this device.

TYPE-*term*
specifies the terminal type to be associated with the specified LNIATA, where *term* is a 2-digit hexadecimal number indicating the type of terminal referenced by the LNIATA or one of the following values:

IPBRIDGE
IP Bridge.

MATIPA
MATIP Type-A conversational device.

MATIPH
MATIP Type-A host-to-host device.

MATIPB
MATIP Type-B device.

MQ
MQ Bridge device.

NOTCPIP
No longer associated with TCP/IP middleware.

DEFine
defines the IP address associated with an LNIATA or defines the host name. The host name must not be already defined.

DELete
deletes a host name or ASCUs. The host must be already defined.

DISplay
displays session information associated with an LNIATA and session characteristics for a host name. If the ASCU parameter is also specified, the ASCU list associated with the host name will be displayed.

STArt
starts the MATIP Type-A and Type-B listeners, and activates MATIP support.

STOp
stops the MATIP Type-A and Type-B listeners, and closes all active MATIP sessions.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMATP HELP
ZMATP ?

Examples

The following example displays the status of a session using LNIATA FE1401.

```
User:   ZMATP DISPLAY FE1401

System: MATP0006I 13.26.55 DISPLAY MATIP SESSION

         LNIATA:                 FE1401
         SOCKET NUMBER:            251
         IP ADDRESS:             9.117.198.103
         TRAFFIC TYPE:           TYPE A
         ENCODING:               EBCIDIC
         MULTIPLEX:             2-BYTE ASCUS
         HEADER:                4-BYTE HEADER
         PRESENTATION:          1024C
         H1H2:                  3F
         NUMBER OF ASCUS:       10
```

The following example adds IP address 9.117.107.92 to the WGTA entry, and changes the terminal type from A5 to 43 for LNIATA FE0412.

```
User:   ZMATP DEFINE FE0412 IP-9.117.107.92 TYPE-MATIPH

System: MATP0001I 13.26.55 IP ADDRESS DEFINED FOR LNIATA-FE0412
                           OLD            NEW
        TERMINAL TYPE  -   A5             43
        IP ADDRESS     -   NONE           9.117.107.92

        END OF DISPLAY
```

The following example removes the IP address associated with LNIATA FE0412 and resets the terminal type to A5.

```
User:   ZMATP ALTER FE0412 IP-CLEAR TYPE-A5

System: MATP0002I 13.26.55 INFORMATION ALTERED FOR LNIATA-FE0412
                           OLD            NEW
        TERMINAL TYPE  -   43             A5
        IP ADDRESS     -   9.117.107.92   NONE

        END OF DISPLAY
```

The following example activates MATIP support.

```
User:   ZMATP START

System: MATP0007I 13.26.55 MATIP STARTED SUCCESSFULLY
```

The following example deactivates MATIP support.

```
User:   ZMATP STOP

System: MATP0008I 13.26.55 MATIP STOPPED
```

The following example defines a remote host name.

```
User:   ZMATP DEF RHOSTN-ABC.DEFG.HIJK.LMN

System: MATP0021I 13.26.55 HOST NAME DEFINED
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: NONE
```

The following example changes a host name with 2-byte ASCUs.

```
User:   ZMATP ALTER RHOSTN-ABC.DEFG.HIJK.LMN ASCU-4141/5533

System: MATP0020I 13.26.55 HOST NAME ALTERED
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: 4141 5533
```

The following example deletes a remote host name.

```
User:   ZMATP DEL RHOSTN-ABC.DEFG.HIJK.LMN

System: MATP0018I 13.26.55 INFORMATION DELETED
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: NONE
```

The following example deletes 2-byte ASCUs from a host name.

```
User:   ZMATP DEL RHOSTN-ABC.DEFG.HIJK.LMN ASCU-4141/5533/2345/6162/3456/7171

System: MATP0018I 13.26.55 INFORMATION DELETED
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: 4141 5533 2345 6162 3456 7171
```

The following example adds 4-byte ASCUs to a host name.

```
User:   ZMATP ADD RHOSTN-ABC.DEFG.HIJK.LMN ASCU-41552345/61347181/F1231111

System: MATP0017I 13.26.55 ASCUS ADDED FOR HOST NAME
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: 41552345 61347181 F1231111
```

The following example displays the ASCU list associated with a host name.

```
User:   ZMATP DISPLAY ASCU RHOSTN-ABC.DEFG.HIJK.LMN

System: MATP0006I 13.26.55 DISPLAY MATIP SESSION
        HOST: ABC.DEFG.HIJK.LMN
        ASCU: 43721294 10398452 30491048 20494451
```

The following example displays the characteristics associated with a host name.

```
User:   ZMATP DISPLAY RHOSTN-ABC.DEFG.HIJK.LMN ASCU

System: MATP0006I 13.26.55 DISPLAY MATIP SESSION
        HOST: ABC.DEFG.HIJK.LMN
        TYPE-A CONVERSATIONAL
         LNIATA:                FE1401
         SOCKET NUMBER:         251
         IP ADDRESS:            9.117.198.103
         TRAFFIC TYPE:          TYPE A
         ENCODING:              EBCIDIC
         MULTIPLEX:             2-BYTE ASCUS
         HEADER:                4-BYTE HEADER
         PRESENTATION:          1024C
         H1H2:                  3F
         NUMBER OF ASCUS:       10

        TYPE B

         NO ACTIVE SESSION

        TYPE-A HOST TO HOST
         LNIATA:                FE1414
         SOCKET NUMBER:         157
         IP ADDRESS:            9.117.198.102
         TRAFFIC TYPE:          H TO H
         ENCODING:              EBCIDIC
         MULTIPLEX:             SINGLE FLOW
         HEADER:                NO HEADER
         PRESENTATION:          1024C
         H1H2:                  76
         FLOW ID:               42

        END OF DISPLAY
```

Related Information

See *TPF Application Programming* and *TPF Data Communications Services Reference* for more information about the terminal address table (WGTA).

ZMCFT ADD–Add a Coupling Facility

Use this command to add a coupling facility (CF) to a processor configuration.

Requirements and Restrictions

You can enter this command only after CF locking restart has been completed.

Format

```
►►──ZMCFT Add── ──cfname── ──┬─sda──────────────┬──────────────────────────►◄
                             └─PROC── ──procid──┘
```

cfname
is the 5- to 8-character alphanumeric name of the CF you want to add. The first character of the CF name must be an alphabetic character.

sda
is any 1- to 4-character hexadecimal symbolic device address (SDA) associated with the CF.

PROC *procid*
specifies a processor whose configuration will be updated, where *procid* is the 1-character alphanumeric processor identifier (ID). The processor ID corresponds to an inactive processor. The CF must have been added previously to the processor from which the command was entered.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZMCFT HELP
ZMCFT ?

Examples

The following example shows a CF named QUEUE, which is attached to SDA 1001, being added to the processor configuration.

```
User:    ZMCFT ADD QUEUE 1001

System:  MCFT0001I 07.59.28 CFMADD - COUPLING FACILITY QUEUE ADDED -
                            2 PATHS EXIST
```

The following example shows a CF being added to another processor configuration.

```
User:    ZMCFT ADD QUEUE PROC C

System:  MCFT0029I 07.59.15 CFMADD - COUPLING FACILITY QUEUE ADDED TO THE
                            CONFIGURATION OF PROCESSOR C
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZMCFT CLEAR–Clear Coupling Facility Structures from a Coupling Facility

Use this command to remove from a coupling facility (CF) the CF structures that are not known to this processor configuration.

Requirements and Restrictions

Wait for CF restart to be completed before entering this command.

Format

```
►►──ZMCFT CLEAR── ──cfname─────────────────────────────────────────►◄
```

cfname
: is the 5- to 8-character alphanumeric name of the CF from which you want to clear CF structures. The first character of the CF name must be an alphabetic character.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

The following example shows CF structures being cleared from the CF named MQ123.

```
User:    ZMCFT CLEAR MQ123

System:  MCFT0035I 07.59.28 CFMCLE - COUPLING FACILITY MQ123 CLEARED
```

Related Information

See *TPF Database Reference* for more information about CF record lock support.

ZMCFT DELETE–Delete a Coupling Facility

Use this command to remove a coupling facility (CF) from the processor configuration.

Requirements and Restrictions

You can enter this command only after CF locking restart has been completed.

Format

```
►►──ZMCFT DElete── ──cfname──┬─────────────────────┬──────────────►◄
                             └─ ──PROC── ──procid─┘
```

cfname
is the 5- to 8-character alphanumeric name of the CF you want to delete from your processor configuration. The first character of the CF name must be an alphabetic character.

PROC *procid*
specifies a processor whose configuration will be updated, where *procid* is the 1-character alphanumeric processor identifier (ID). The processor ID must correspond to an inactive processor.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

The following example shows the CF named QUEUE being deleted from the processor configuration.

```
User:    ZMCFT DEL QUEUE

System:  MCFT0002I 07.59.28 CFMDEL - COUPLING FACILITY QUEUE DELETED
```

The following example shows a CF being deleted from another processor configuration.

```
User:    ZMCFT DEL QUEUE PROC C

System:  MCFT0043I 07.59.28 CFMDEL - COUPLING FACILITY QUEUE DELETED FROM THE
         CONFIGURATION OF PROCESSOR C
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZMCFT DISPLAY–Display Coupling Facility Status

Use this command to display the status of one or more coupling facilities (CFs) in the processor configuration.

Requirements and Restrictions

You can enter this command only after CF locking restart has been completed.

Format

```
                     ┌─ ──ALL──────────────┐
►►──ZMCFT DIsplay────┼─────────────────────┼──────────────────────────────►◄
                     ├─ ──ALL── ──procid───┤
                     └─ ──cfname───────────┘
```

ALL
: displays the CF name and CF identifier (ID) for each CF in a processor configuration if a processor ID is specified or displays the CF name and CF ID for each CF in the processor configuration if a processor ID is not specified.

procid
: is the 1-character alphanumeric processor ID of the processor whose CF information you want to display. If you do not specify a processor ID, information for all CFs in your processor configuration is displayed rather than the information for one specific CF.

  Note: This parameter is ignored on TPF systems that were not generated for a shared data processing system (SDPS).

cfname
: is the 5- to 8-character alphanumeric name of the CF whose status you want to display. The first character of the CF name must be an alphabetic character.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

Information about a specific CF is displayed in the following example.

Note: The (D) following a processor ID indicates that the CF was disabled for the processor. The information about whether the CF was disabled for a processor is only available for the processor from which the ZMCFT DISPLAY command is processed. The (N) following a processor ID indicates the processor has not been through CF locking restart since it was added initially for the processor.

```
User:    ZMCFT DISPLAY SUECF

System:  MCFT0047I 19.29.52 CFMDIS – ZMCFT DISPLAY
         CF NAME - SUECF
         CF ID   - 18000500E2C9D4C4C5E5F0F01C9C2D4
                   C5D6D2C5D4F1F0F0F0F0F0F0F0F0000
         AVAILABLE TO PROCESSORS -
         B C D(D) E(N)
         SDAS FOR THIS PROCESSOR -
         1000 1001 1002 1003
         ALLOCATED STRUCTURES -              TYPE         - CONNECTED PROCESSORS
           C0038                             LIST           B C
           C0039                             LIST           B C
           C0040                             CACHE          B C
         END OF ZMCFT DISPLAY
```

Information about all CFs in the processor configuration is displayed in the following example.

```
User:    ZMCFT DISPLAY ALL

System:  MCFT0004I 19.29.52 CFMDIS – ZMCFT DISPLAY
         CF NAME       CF ID
         --------      -------------
         SUECF         18000500E2E4C3C64040404040404040
                       40404040404040404040404040400000
         LOCKING       18000600E2E3C2C55050505050505050
                       50505050505050505050505050500000
         END OF ZMCFT DISPLAY
```

Information about all CFs added to the processor configuration for processor B is displayed in the following example.

```
User:    ZMCFT DISPLAY ALL B

System:  MCFT0004I 19.29.52 CFMDIS – ZMCFT DISPLAY
         CF NAME       CF ID
         --------      -------------
         LOCKING       18000600E2E3C2C55050505050505050
                       50505050505050505050505050500000
         END OF ZMCFT DISPLAY
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZMCFT ENABLE–Enable a Coupling Facility

Use this command to resume normal operations of a coupling facility (CF) that was active previously, but became inactive when an error occurred.

Requirements and Restrictions

You can enter this command only after CF locking restart has been completed.

Format

```
►►──ZMCFT Enable── ──cfname──────────────────────────────────────────►◄
```

cfname
is the 5- to 8-character alphanumeric name of the CF you want to enable. The first character of the CF name must be an alphabetic character.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

The following example shows normal operations being resumed on a CF that had been disabled.

```
User:    ZMCFT ENABLE QUEUE

System:  MCFT0008I 07.59.15 CFMENA - COUPLING FACILITY QUEUE ENABLED
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZMCFT REMOVE–Remove Processor Connections to a Coupling Facility

Use this command to remove all inactive connections a processor has to a coupling facility (CF) structure.

Requirements and Restrictions

- You can enter this command only after CF restart has been completed.
- You can only remove inactive connections that a processor has to a CF structure. If active connections are present, the command is rejected.

Format

```
►►──ZMCFT REMove── ──STRucture── ──strname── ──CF── ──cfname──────────────────►

►──┬──────────────────────┬──────────────────────────────────────────────────►◄
   └─ ──PROC── ──procid───┘
```

STRucture *strname*
specifies the structure from which you want to remove the processor connections, where *strname* is a 1- to 16-character structure name.

CF *cfname*
specifies a CF that contains the structure, where *cfname* is the 5- to 8-character alphanumeric name of the CF. The first character of the CF name must be an alphabetic character.

PROC *procid*
specifies a processor whose connections will be removed, where *procid* is the 1-character alphanumeric processor identifier (ID). The processor ID corresponds to an inactive processor. The CF must have been added previously to the processor from which the command was entered.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

The following example shows the connection for processor B from which the command was entered being removed from a structure named ROUTER on the CF named MQ123.

```
User:    ZMCFT REMOVE STRUCTURE ROUTER CF MQ123

System:  MCFT0048I 07.59.15 CFMREM - CONNECTION TO STRUCTURE ROUTER ON COUPLING
                                     FACILITY MQ123 WAS REMOVED FOR PROCESSOR B
```

The following example shows the connection for an inactive processor named D being removed from a structure named Q2000 on the CF named REALCF1.

```
User:    ZMCFT REMOVE STRUCTURE Q2000 CF REALCF1 PROC D

System:  MCFT0048I 07.59.15 CFMREM - CONNECTION TO STRUCTURE Q2000 ON COUPLING
                                     FACILITY REALCF1 WAS REMOVED FOR PROCESSOR D
```

Related Information

- See *TPF Database Reference* for more information about CF support.
- See “ZMCFT ADD–Add a Coupling Facility” on page 754 for more information about the ZMCFT ADD command.

ZMCFT RESETLOCK–Reset a Coupling Facility Lock

Use this command to reset a coupling facility (CF) lock to an available state. An *available state* is the state a CF is in when all CF commands are processed normally. The CF lock is used to serialize operations on the CF. Use this command only if another function that sets and resets the CF lock ends in error.

Requirements and Restrictions

None.

Format

```
►►──ZMCFT RESetlock── ──cfname──────────────────────────────────────────►◄
```

cfname
is the 5- to 8-character alphanumeric name of the CF whose CF lock you want to reset. The first character of the CF name must be an alphabetic character.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMCFT HELP
ZMCFT ?

Examples

The following example shows the resetting of the CF lock on the CF named QUEUE.

```
User:    ZMCFT RESETLOCK QUEUE

System:  MCFT0024I 07.59.14 CFMRES - CF LOCK WAS RESET FOR COUPLING
                            FACILITY QUEUE
```

Related Information

See *TPF Database Reference* for more information about CF support.

ZMCHR–Convert FARF File Address to Extended MCHR Format Address

Use this command to convert a given FARF file address to the physical disk address in extended MCHR format.

Requirements and Restrictions

None.

Format

```
►►──ZMCHR── ──fileaddr──────────────────────────────────────────────►◄
```

fileaddr
is an 8- or 16-digit hexadecimal file address.

Note: The FARF address must be compatible with the dispense modes in the current migration stage, FARF3/4 or FARF4/5, because it will be interpreted as an address in one of the two existing formats. This does not apply to FARF6 file addresses, but does apply to 4x4 format file addresses.

Additional Information

None.

Examples

The specified FARF address is converted to MCHR format in the following example. The 14-character string in the system response is interpreted as follows:

- Characters 1–4 (in this example, 0047) represent the symbolic module number.
- Characters 5–8 (in this example, 0029) represent the 2–byte cylinder number.
- Characters 9–12 (in this example, 0000) represent the 2–byte head number.
- Characters 13–14 (in this example, 05) represent the record number with control bits to indicate record size and duplication.

```
User:   ZMCHR 00000000F4024011

System: MCHR0001I 13:58:35 00470029000005 DUPE 4K
```

Related Information

See *TPF Database Reference* for more information about file address formats.

ZMCPY ABORT–File Copy Abort

Use this command to stop ZMCPY ALL or ZMCPY UP processing.

Requirements and Restrictions

None.

Format

```
►►──ZMCPY Abort── ──┬─ALL─┬──────────────────────────────────────────►◄
                    └─nn──┘
```

ALL
stops all module copies on all processors in the processor complex.

nn is the slot number of a module copy. The specified module copy stops if it is active on the current subsystem.

Additional Information

Entering ZMCPY ABORT to end the ZMCPY ALL command takes the *destination* DASD offline.

Examples

In the following example, the ZMCPY function that was previously running is stopped.

```
User:   ZMCPY ABORT 1

System: MCPY0203T 12.23.10 MCPY 01  COPY PROCEEDING TO ABORT            '
        MCPY0236T 12.23.30 MCPY 01  ALL FILE COPY ABORTED
```

Related Information

- See “ZMCPY ALL–All File Copy” on page 765 for more information about the *destination* DASD.

ZMCPY ALL–All File Copy

Use this command to copy an entire DASD from one online DASD to another.

Requirements and Restrictions

None.

Format

```
►►──ZMCPY ALL── ──origin── ──destination──┬─────────────────┬──────────────►◄
                                          └── ──TOKEN-token─┘
```

origin
is the 3- to 4-digit hexadecimal device address of the DASD that you want to copy.

destination
is the 3- to 4-digit hexadecimal device address of the DASD to which you want to copy.

TOKEN-*token*
is a 1- to 4-character user token that is passed to the UCPY user exit.

Additional Information

- You can use this command, in any system state, to copy data from a DASD that is experiencing hardware errors.
- The device types of the DASDs are verified to ensure that they are the same and that the configurations are compatible before the specified DASD is copied.

  Notes:
  1. Copying a multiple-density device to a lesser-density device can produce unpredictable results.
  2. Copying a prime or duplicate DASD to a DASD on the same channel or control unit (CU) can degrade performance.
- Before entering the ZMCPY ALL command, the *destination* DASD must have a volume serial number (VSN) of *zz*9999, where*zz* are the first two characters of the *origin* DASD.
- When the ZMCPY ALL command ends, the *destination* DASD has the volume serial number that was previously on the *origin* DASD, and the *origin* DASD now has a volume serial number of *zz*9999.
- Enter the ZMCPY ABORT command to stop the ZMCPY ALL function.

Examples

In the following example, DASD 05E2 is copied to DASD 05E1.

```
User:   ZMCPY ALL 05E2 05E1

System: MCPY0209I 08.44.23 MCPY 01  COPY STARTING...
        MCPY0218I 08.44.25 MCPY 01  ALL FILE COPY STARTED
        MCPY0245I 08.44.25 MCPY 01  IPL RECORDS SUCCESSFULLY COPIED
```

Related Information

None.

ZMCPY DOWN–Module Down Processing

Use this command to change the status of a direct access storage device (DASD) from online to offline. This function updates the module file status table (MFST) and cross reference table to reflect the configuration change.

Requirements and Restrictions

None.

Format

```
►►──ZMCPY Down── ──module── ──device──────────────────────────────────────►◄
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time packs or the pseudo module number for general files.

device
: is a 3- to 4-digit hexadecimal device address.

Additional Information

None.

Examples

The status of the specified DASD is allowed to change from online to offline in the following example.

```
User:   ZMCPY D 4A EE0

System: MCPY0294I 14.46.36 OK MOD 04A OFF DEVICE 0EE0
```

Related Information

See *TPF Database Reference* for more information about module down processing.

ZMCPY PAUSE–Pause File Copying

Use this command to pause ZMCPY ALL and duplicate file copying.

Requirements and Restrictions

You can start the ZMCPY PAUSE command from any active processor while a copy function is in progress.

Format

```
►►──ZMCPY Pause── ──┬─ALL─┬──────────────────────────────────────────►◄
                    └─nn──┘
```

ALL
pauses all module copies on all processors in the processor complex.

nn pauses the specified module copy if it is active on the current subsystem.

Additional Information

None.

Examples

In the following example, all copy functions are paused.

```
User:   ZMCPY PAUSE ALL

Screen: MCPY0268I 15.28.31 MCPY  -  COPY PAUSING
        MCPY0261I 15.28.31 MCPY 01  COPY PAUSED
        MCPY0261I 15.28.31 MCPY 02  COPY PAUSED
```

Related Information

None.

ZMCPY RESTART–File Copy Restart

Use this command to restart ZMCPY ALL or ZMCPY UP processing.

Requirements and Restrictions

You can enter this command only if a catastrophic system error occurred while the ZMCPY function was in progress or if the operator had previously paused the copy. If ZMCPY ALL or ZMCPY UP processing is aborted by the operator or forced to abort by the copy program, it cannot be restarted.

Format

```
►►──ZMCPY Restart── ──nn──────────────────────────────────────────►◄
```

nn is the slot number of a module copy. The specified module copy restarts if it is active on the current processor and subsystem.

Additional Information

None.

Examples

Copy function number 1 is started again in the following example.

```
User: ZMCPY RESTART 1

System: MCPY0209I 15.29.38 MCPY 01  COPY STARTING...
        MCPY0218I 15.29.38 MCPY 01  ALL FILE COPY STARTED
        MCPY0245I 15.29.38 MCPY 01  IPL RECORDS SUCCESSFULLY COPIED
```

Related Information

None.

ZMCPY SET–Set Multicopy Limits

Use this command to display or set the maximum number of module copies that can run concurrently in a processor complex, on a channel, on a control unit, or on a processor.

Requirements and Restrictions

None.

Format

```
►►──ZMCPY SET──┬─ ──CHannel── ──maxchan────┬─────────────────────────────────►◄
               ├─ ──COmplex── ──maxplex────┤
               ├─ ──CU── ──maxcu───────────┤
               ├─ ──PRocessor── ──maxproc──┤
               └─ ──Display────────────────┘
```

CHannel
sets the maximum number of module copies that can run concurrently on a channel.

maxchan
is a decimal number that is equal to 10 or less.

COmplex
sets the maximum number of module copies that can run concurrently in a processor complex.

maxplex
is a decimal number that is equal to 20 or less.

CU
sets the maximum number of modules that can participate in copy functions concurrently on a control unit.

maxcu
is a decimal number that is equal to or less than 5.

PRocessor
sets the maximum number of module copies that can run concurrently on a processor.

maxproc
is a decimal number that is equal to or less than 10.

Display
displays the maximum number of module copies that can run concurrently in a processor complex, on a channel, on a control unit, or on a processor.

Additional Information

None.

Examples

In the following example, the maximum number of copies that can occur concurrently off a control unit is set to 4.

```
User:   ZMCPY SET CU 4

System: MCPY0276I 13.21.28 MCPY  -  MAXIMUM NUMBER OF COPIES ALLOWED:
              COMPLEX   = 20
              CHANNEL   = 10
              CU        =  4
              PROCESSOR = 10
```

Related Information

None.

ZMCPY STATUS–Display File Copy Status

Use this command to display the status of devices that are being copied. You can start the ZMCPY STATUS command from any active processor while a copy function is in progress.

Requirements and Restrictions

None.

Format

```
              ┌─ ──ALL─┐
►►──ZMCPY Status─┤        ├──────────────────────────────────────────────────►◄
              └─ ──nn──┘
```

ALL
displays the status of all module copies on all processors in the processor complex.

nn is the slot number of a module copy. The specified module copy is displayed.

Additional Information

None.

Examples

In the following example, status is displayed for copy number 1 of an ALL FILE COPY function.

```
User:   ZMCPY STATUS 1

System: MCPY0290I 13.41.01 COPY  1 - ALL FILE COPY ACTIVE   ON PROC B
        FROM  DVC  MOD    TO    DVC  MOD
             0EE5  047         0EE6  0DE
        CURRENT CCHH   0049 00
```

Related Information

None.

ZMCPY UP–Module Up Processing

Use this command to change the status of a direct access storage device (DASD) from offline to online. This function updates the module file status table (MFST) and cross reference table to reflect the configuration change. A duplicate update also starts when the DASD is brought online.

Requirements and Restrictions

None.

Format

```
►►──ZMCPY Up── ──module── ──device──┬──────────────┬──────────────────────►◄
                                    └─ ──TOKEN-token┘
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time packs or the pseudo module number for general files.

device
: is a 3- to 4-digit hexadecimal device address.

TOKEN-*token*
: is a 1- to 4-character user token that is passed to the UCPY user exit.

Additional Information

None.

Examples

The status of the specified DASD is allowed to change from offline to online in the following example.

```
User:   ZMCPY UP 4A EE8

System: MCPY0209I 09.18.43 MCPY 01  COPY STARTING...
        MCPY0218I 09.18.43 MCPY 01  DUPE UPDATE STARTED
        MCPY0245I 09.18.43 MCPY 01  IPL RECORDS SUCCESSFULLY COPIED
        MCPY0247I 09.24.37 MCPY 01  ENTERING EOJ
        MCPY0228T 09.24.37 MCPY 01  DUPE UPDATE COMPLETED
```

Related Information

See *TPF Database Reference* for more information about module up processing.

ZMEAS–Start Data Collection

Use this command to start data collection. You can start data collection in continuous mode or sampling mode.

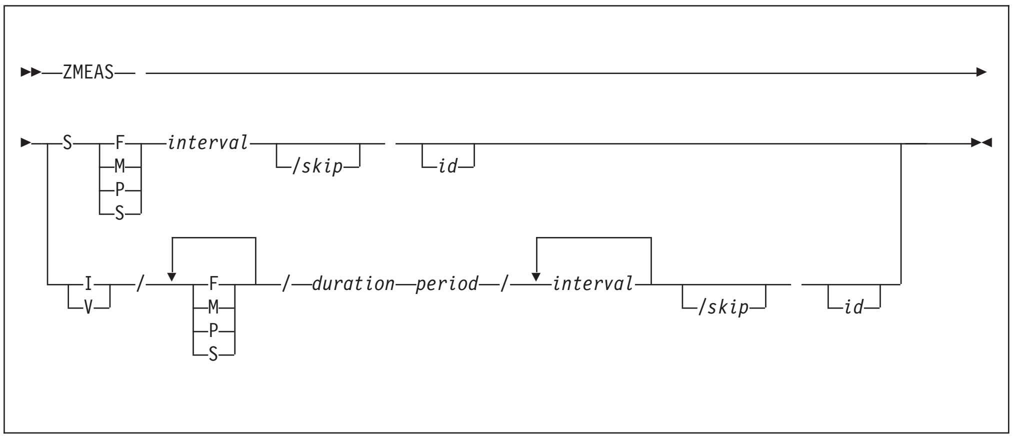
In *continuous mode*, all 4 of the collectors are run one after the other to gather summary performance data, and then one collector is run for a specified interval of time to collect detailed performance data.

In *sampling mode*, both summary and detailed performance data are gathered in turn by one or more specified collectors. These collectors are run one after the other in a specified order for specified collection intervals.

Requirements and Restrictions

The TPF system must be above 1052 state and cannot be cycling.

Format



S starts data collection in continuous mode.

I starts data collection in sampling mode with identical collection intervals for each collector.

V starts data collection in sampling mode with individual collection intervals for each collector.

F starts the file collector, which collects file and virtual file address (VFA) performance data.

M starts the message collector, which collects message performance data.

P starts the program collector, which collects program performance data.

S starts the system collector, which collects a cross section of system performance data.

interval
is the 2-digit number of seconds for the collection interval.

Note: If you specified the V parameter, specify an individual collection interval for each collector that you specified. The order of the collection intervals corresponds with the order of the collectors.

duration
is the 2-digit number of minutes that data collection is run (when it is started in sampling mode).

period
is the 2-digit number of seconds for the sampling period. This value must be at least 3 seconds more than the sum of the collection intervals that you specified.

skip
is the 4-digit number of collection events to skip before sampling an event. A small number indicates that fewer events are skipped, which increases the amount of data on the RTC tape and impacts the performance of the TPF system. A large number indicates that more events are skipped, which lessens the impact on system performance.

Note: The SKIP parameter on the DATACO macro (which has a default of 99) sets the default value of the *skip* parameter on the ZMEAS command. If the parameter is changed on the DATACO macro, it will also be changed on the *skip* parameter of the ZMEAS command.

id is the 1-character alphanumeric CPU ID of the processor on which complexwide data is to be exclusively collected. If you omit this variable, complexwide data can be collected by any processor in the TPF system. See “Loosely Coupled Considerations for Running Data Collection and Reduction” on page 1474 for more information about collecting complexwide data.

Note: This variable is valid only for loosely coupled TPF complexes. The specified processor must be system generated and active, and the basic subsystem (BSS) cannot be cycling or be below CRAS state on the processor.

Additional Information

Sampling mode has less of an impact on system performance than does continuous mode because of the sampling mode gaps. Larger gaps lessen the impact on system performance even more.

Examples

Data collection is started in continuous mode in the following example. Data is continuously collected for 300 5-second intervals.

```
User:   ZMEAS SS05

System: MEAS0001I 11.58.40 STARTING
```

Data collection is started in sampling mode in the following example. The system collector runs first, followed by the program collector, the file collector, and the message collector. Each collector is run, in turn, for 5 seconds of the 23-second sampling period. This 23-second sampling period repeats for 30 minutes.

```
User:   ZMEAS I/SPFM/3023/05

System: MEAS0001I 11.58.40 STARTING
```

Data collection is started in sampling mode in the following example. The system collector runs first for 10 seconds and then the message collector runs for 15 seconds. The total sampling period lasts for 28 seconds, which implies a 3-second gap after the collection cycle. The sampling period repeats for 30 minutes. After data collection samples an event, 9999 events are skipped before another event is sampled.

```
User:   ZMEAS V/SM/3028/1015/9999

System: MEAS0001I 14.32.00 STARTING
```

Related Information

See “System Performance and Measurement” on page 1457 and *TPF System Performance and Measurement Reference* for more information about data collection.

ZMEAS END–End Data Collection

Use this command to end data collection.

Requirements and Restrictions

None.

Format

```
►►──ZMEAS END────────────────────────────────────────────────────────►◄
```

Additional Information

- If data collection is running in continuous mode, it does not end until after the current collection interval is completed.
- If data collection is running in sampling mode, it does not end until after the current sampling period is completed.

Examples

Data collection is ended in the following example.

```
User:   ZMEAS END

System: MEAS0002I 07.13.55 TERMINATION REQUESTED
        MEAS0003I 07.14.13 TERMINATED
        COTG0080A 07.14.13 TOFF BSS    REMOVE RTC FROM DEVICE 421
                           VSN A00039 G0002 S0001 D38K   SL  NOBLK  NOCOMP
```

Related Information

See "System Performance and Measurement" on page 1457 and *TPF System Performance and Measurement Reference* for more information about data collection.

ZMEAS Q–Query or Change Tape Threshold

Use this command to query or change the data collection tape threshold. The *data collection tape threshold* is the number of tape writes that can be queued before data collection quiesces.

Requirements and Restrictions

None.

Format

```
►►──ZMEAS Q──┬─────────────┬──────────────────────────────────────►◄
             └──threshold──┘
```

threshold
is the 4-digit data collection tape threshold.

Note: If you do not specify a threshold value, the current tape threshold is displayed.

Additional Information

- The data collection tape threshold is reset to the default value if an initial program load (IPL) is performed.
- The default data collection tape threshold is defined by the value assigned to RTCQUEUE in the DC0DC DSECT.

Examples

The current data collection tape threshold is displayed in the following example.

```
User:   ZMEAS Q

System: MEAS0029I 07.17.06   64 CURRENT   64 PREVIOUS   64 ORIGINAL TAPE THRESHOLDS
```

The data collection tape threshold is changed in the following example.

```
User:   ZMEAS Q0123

System: MEAS0029I 07.17.06  123 CURRENT   64 PREVIOUS   64 ORIGINAL TAPE THRESHOLDS
```

Related Information

See “System Performance and Measurement” on page 1457 and *TPF System Performance and Measurement Reference* for more information about data collection.

ZMEAS RESET–Reset Data Collection

Use this command to reset data collection when it fails because of a non-catastrophic error.

Requirements and Restrictions

Do not use this command if data collection stalls because the system task dispatcher is shut down.

Format

```
►►──ZMEAS RESET──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

Data collection is ended in the following example.

```
User:   ZMEAS RESET

System: ZMEAS RESET
        MEAS0028I 07.17.06 DATA COLLECTION HAS BEEN RESET
        COTG0080A 07.17.07 TOFF BSS    REMOVE RTC FROM DEVICE 421
                           VSN A00039 G0003 S0001 D38K   SL  NOBLK  NOCOMP
```

Related Information

See “System Performance and Measurement” on page 1457 and *TPF System Performance and Measurement Reference* for more information about data collection.

ZMIGR–32-Way Loosely Coupled Migration

Use this command to run 32-way loosely coupled processor support migration functions, which include the following:

- Initialization or conversion of fixed records
- A display of record or processor status
- Fallback to previous fixed record formats.

Requirements and Restrictions

- All processors must be running 32-way loosely coupled processor support before CONVERT or FALLBACK parameter requests can be serviced.
- CONVERT and FALLBACK parameter requests are serviced only if eight or fewer processors are generated in the loosely coupled complex.
- For CTKI, the CONFIG SIP macro in the system generation deck must have parameter CTKI32LC specified to reflect the current state of keypoint I (CTKI). After conversion, if CTKI is to be generated, you must code CTKI32LC=YES on the CONFIG macro. If a FALLBACK is then performed, you must code CTKI32LC=NO. This ensures that the CTKI indicator, IC032LC, is not lost across a system generation.
- For CTKC, the CRASTB SIP macro in the system generation deck must have parameter CTKC32LC specified to reflect the current state of CTKC. After conversion, you must code CTKC32LC=YES on the CRASTB macro. If a FALLBACK is then performed, you must code CTKC32LC=NO. This ensures that the CTKC 32-way loosely coupled processor support flag, CK8LC32, is not lost across a system generation.

Format

```
►►──ZMIGR── ──┬─CTKC── ──┬─Convert──┬──┬──────────────────────────────►◄
              │          ├─Fallback─┤  │
              │          └─Status───┘  │
              ├─CTKI── ──┬─Convert──┬──┤
              │          ├─Fallback─┤  │
              │          └─Status───┘  │
              ├─Keypoint── ──┬─Convert──┬┤
              │              ├─Fallback─┤│
              │              └─Status───┘│
              ├─Mpif── ──┬─Convert──┬──┤
              │          ├─Fallback─┤  │
              │          └─Status───┘  │
              ├─Processor── ──Status───┤
              └─Recoup── ──┬─Convert──┬┘
                           ├─Fallback─┤
                           ├─Init─────┤
                           └─Status───┘
```

CTKC
: performs the function specified by the keyword parameter on the computer room agent set (CRAS) information in keypoint C (CTKC). The CTKC parameter is valid only when entered under the basic subsystem (BSS).

The CRAS table information is converted to or restored from 32-way loosely coupled format.

CTKI
perfroms the function specified by the keyword parameter on the subsystem state table (IC0SST) in CTKI. The CTKI parameter is valid only when entered under the basic subsystem (BSS).

The subsystem state table IC0SST is converted to or restored from #CN1ST fixed file records.

Keypoint
performs the function specified by the keyword parameter on the keypoint control records.

The keypoint status information in #KBA ordinal 0, #KSA1 – #KSA8 ordinal 0, and #IBMM4 ordinal 54 (#KPTCNTL) is converted to, or restored from, 32-way loosely coupled format.

Mpif
performs the function specified by the keyword parameter on ordinals 2 and 3 of record type #PDREC. The MPIF parameter is valid only when entered under the BSS and only if the MPIF feature is installed on the TPF system.

#PDREC ordinals 2 and 3 are copied to #IBMM4 records as a save area or are restored to the #PDREC records from the #IBMM4 records.

Processor
displays the status of 32-way loosely coupled processor support for each active processor in the complex.

Recoup
performs the function specified by the keyword parameter on the 1052 FC33 records.

Convert
converts the table or record to 32-way loosely coupled format or copies the record its new location for 32-way loosely coupled processor support.

Fallback
restores the table or record to the format or location that it had before converting to 32-way loosely coupled processor support.

Init
initializes the #RC8RFS (FC33) records, ordinals 65 and higher.

Note: This parameter is valid only with the RECOUP parameter.

Status
displays the 32-way loosely coupled processor support migration status for the table, record, or all active processors.

STATUS parameter requests are processed in all configurations, including the following:

- Processors running with and without 32-way loosely coupled processor support in the same loosely coupled complex
- Configurations in loosely coupled complexes with eight or fewer processor or more than eight processors.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMIGR HELP
ZMIGR ?

Examples

The following example shows the CTKC conversion function:

```
User:   ZMIGR CTKC CONVERT

System: MIGR0040I hh.mm.ss CTKC CONVERSION COMPLETED
```

The following example shows the CTKC fallback function:

```
User:   ZMIGR CTKC FALLBACK

System: MIGR0041I hh.mm.ss CTKC FALLBACK COMPLETED
```

The following example shows the CTKC status function:

```
User:   ZMIGR CTKC STATUS

System: MIGR0042I hh.mm.ss CTKC IS NOT CONVERTED

    or: MIGR0043I hh.mm.ss CTKC IS CONVERTED
```

The following example shows the CTKI conversion function:

```
User:   ZMIGR CTKI CONVERT

System: MIGR0001I hh.mm.ss CTKI CONVERSION COMPLETED, SUBSYSTEM STATE
                           TABLE HAS BEEN MOVED FROM CTKI TO FIXED FILE
                           RECORD TYPE #CN1ST
```

The following example shows the KEYPOINT conversion function:

```
User:   ZMIGR KEYPOINT CONVERT

System: MIGR0009I hh.mm.ss #KEYPT CONVERSION COMPLETED
        MIGR0009I hh.mm.ss #KBA CONVERSION COMPLETED
        MIGR0009I hh.mm.ss #KSA1 CONVERSION COMPLETED
        . . .
        MIGR0009I hh.mm.ss #KSAn CONVERSION COMPLETED
        MIGR0010I hh.mm.ss KEYPOINT CONVERSION REQUEST COMPLETED

User:   ZMIGR KEYPOINT STATUS

System: MIGR0013I hh.mm.ss #KEYPT CONVERTED
        MIGR0013I hh.mm.ss #KBA CONVERTED
        MIGR0013I hh.mm.ss #KSA1 CONVERTED
        . . .
        MIGR0013I hh.mm.ss #KSAn CONVERTED
        MIGR0010I hh.mm.ss KEYPOINT STATUS REQUEST COMPLETED
```

The following example shows the KEYPOINT fallback function:

```
User:   ZMIGR KEYPOINT FALLBACK

System: MIGR0011I hh.mm.ss #KEYPT FALLBACK COMPLETED
        MIGR0011I hh.mm.ss #KBA FALLBACK COMPLETED
        MIGR0011I hh.mm.ss #KSA1 FALLBACK COMPLETED
        . . .
        MIGR0011I hh.mm.ss #KSAn FALLBACK COMPLETED
        MIGR0010I hh.mm.ss KEYPOINT FALLBACK REQUEST COMPLETED

User:   ZMIGR KEYPOINT STATUS

System: MIGR0014I hh.mm.ss #KEYPT NOT CONVERTED
        MIGR0014I hh.mm.ss #KBA NOT CONVERTED
        MIGR0014I hh.mm.ss #KSA1 NOT CONVERTED
        . . .
        MIGR0014I hh.mm.ss #KSAn NOT CONVERTED
        MIGR0010I hh.mm.ss KEYPOINT STATUS REQUEST COMPLETED
```

The following example shows the processor status function:

```
User:   ZMIGR PROCESSOR STATUS

System: MIGR0021I hh.mm.ss 32-WAY LC MIGRATION STATUS
            PROCESSOR        STATUS
            ---------      ----------
                B          MIGRATED
                C          UNMIGRATED
                D          INACTIVE
                E          INACTIVE   _
                Z          INACTIVE
                0          INACTIVE
        END OF DISPLAY+
```

The following example shows the RECOUP conversion function:

```
User:   ZMIGR RECOUP INIT

System: MIGR0036I hh.mm.ss ALL NEW FC33 RECORDS INITIALIZED

User:   ZMIGR RECOUP CONVERT

System: MIGR0030I hh.mm.ss RECOUP CONVERSION COMPLETED

User:   ZMIGR RECOUP STATUS

System: MIGR0032I hh.mm.ss RECOUP MIGRATION STATUS -
                           RECOUP CONVERSION COMPLETED
```

Related Information

See *TPF Migration Guide: Program Update Tapes* for more information about 32-way loosely coupled processor support migration.

ZMODE–Change Dispensing Mode

Use this command to change the dispensing mode in a particular migration stage. For example, stage FARF3/4 can dispense FARF3 or FARF4 addresses.

Requirements and Restrictions

None.

Format

```
►►──ZMODE── ──┬─3─┬──────────────────────────────────────────────────►◄
              ├─4─┤
              ├─5─┤
              ├─6─┤
              └─D─┘
```

3 dispenses FARF3 addresses.

4 dispenses FARF4 addresses.

5 dispenses FARF5 addresses.

6 activates FARF6 8-byte file addressing. FARF6 is independent of FARF3, FARF4, or FARF5 and can be used concurrently.

D displays the current FARF dispense mode.

Additional Information

None.

Examples

The dispensing mode is changed to FARF3 in the following example.

```
User:   ZMODE 3

System: MODE0009I 09.27.38 IN STAGE FARF3/4, DISPENSING MODE IS FARF3
        FARF6 ADDRESSING IS NOT AVAILABLE
```

In the following example, FARF6 addressing is activated for the first time.

Note: In a loosely coupled complex, the MODE0011I message will also be displayed on all other processors in the complex.

```
User:   ZMODE 6

System: MODE0012I 16.49.26 KEYPOINT 9 UPDATED
        CSMP0097I 16.49.26 CPU-B SS-BSS  SSU-HPN  IS-01
        MODE0011I 16.49.26 SYSTC BIT SB8BFAD SET ON ALL I-STREAMS
        CSMP0097I 16.49.26 CPU-B SS-BSS  SSU-HPN  IS-01
        MODE0009I 16.49.26 IN STAGE FARF4/5, DISPENSING MODE IS FARF4
        FARF6 ADDRESSING IS AVAILABLE
```

Related Information

See *TPF Database Reference* for more information about FARF addressing.

ZMPIF DEFINE DEVICE–Define a MPIF Device

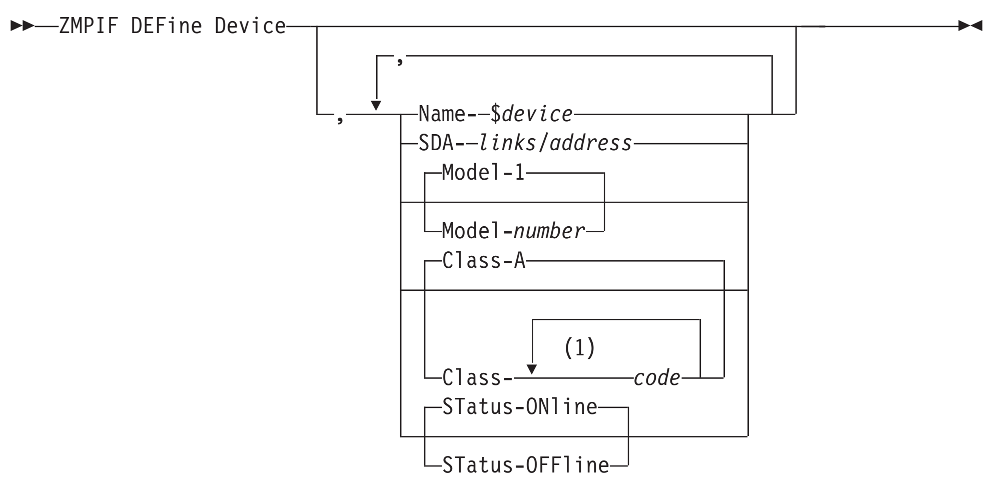
Use this command to do the following:

- Define a new MPIF device
- Change an existing MPIF device definition
- Display information about the MPIF devices that are defined for the next initial program load (IPL).

Requirements and Restrictions

- New and changed device definitions do not take effect until the next IPL.
- You can define or change device definitions only after MPIF restart is completed. Before and during MPIF restart, you can use this command only to display information about the definitions.
- Enter the ZMPIF SET CLASS command before you define any MPIF devices for that class.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - If the command is entered on a processor that does not include 32-way loosely coupled processor support, any definitions that are entered must be reentered when 32-way loosely coupled processor support is installed on the processor.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any definitions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor. If you restore 32-way loosely coupled processor support after falling back to the previous level, the new definitions are also restored.

Format



Notes:

1 Separate each set of 8 class codes with the / character.

Name-$*device*
: is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

  Note: You must specify this parameter when you define a new device or change an existing device definition. The device name that you specify cannot already be defined as a path name.

SDA-*links/address*
: is the addressing range for the device, where *links* is an even number of physical links (NL) and *address* is a 4-digit hexadecimal starting symbolic device address (SDA).

  A 3088 device can have from 2–64 physical links. An Enterprise Systems Connection (ESCON) channel in CTC mode can have from 2–512 physical links.

  Note: You must specify this parameter when you define a new device.

Model-*number*
: is a 1- to 2-character model number for the device.

  The valid model numbers for 3088 devices are 1, 1A, and 2. The valid model number for an ESCON channel in CTC mode is C.

Class-*code*
: is a 1-character alphanumeric code that identifies the path class for the device. You can specify as many as 36 different class codes.

STatus
: specifies the IPL status for the device, where:

  ONline
  : starts the device when you perform an IPL.

OFFline
does not start the device when you perform an IPL. (You must enter the ZMPIF START command to start an offline device.)

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF DEFINE HELP**.
- You can specify the parameters for this command in any order.
- To define a new device, you must specify both the NAME and SDA parameters.
- To change an existing device definition, you must specify the NAME parameter and any other parameters that you want to change.
- Enter **ZMPIF DEFINE DEVICE** (with no parameters) to display the devices that are defined for the next IPL.
- To display the device definition for a specific device, specify only the NAME parameter.
- To change the name of a device, enter the ZMPIF DELETE command to delete the device and then enter the ZMPIF DEFINE DEVICE command to redefine the device.
- To display information about the devices that are currently defined, enter **ZMPIF DISPLAY DEVICE**.

Examples

The following information is displayed in the examples:

DEV-NAME
is the name of the device.

MOD
is the model number of the device.

NL
is the number of physical links defined for the device.

SDA
is the symbolic device address range for the device.

STATUS
is the status of the device, which can be:

ONLINE
Device is started when an IPL is performed.

OFFLINE
Device is not started when an IPL is performed.

CLASS
is the path classes assigned to the device.

Information about the devices that are defined for the next IPL is displayed in the following example.

```
User:   ZMPIF DEF DEV

System: MPIF0018I 06.56.47 DEVICE PARAMETERS DEFINED
        DEV-NAME MOD  NL    SDA    STATUS  CLASS
        $CTOC01  C    16 0900/090F ONLINE  ABC
        $CTOC02  C    16 0A00/0A0F ONLINE  ABC
        $CTOC03  2     4 0388/038B ONLINE  ABC
        $CTOC04  2     4 03A8/03AB ONLINE  ABC
        $CTOC05  2     8 1388/138F ONLINE  ABC
        $CTOC06  2    20 13A8/13BB ONLINE  ABC
        $CTOC07  2     4 238C/238F ONLINE  ABC
        $CTOC08  2     4 23B8/23BB ONLINE  ABC
        $CTOC09  2    20 33A4/33B7 ONLINE  ABC
        $CTOC10  2    20 43A0/43B3 ONLINE  ABC
        $CTOC11  2    16 53A0/53AF ONLINE  ABC
        END OF DISPLAY
```

A new device is defined in the following example.

```
User:   ZMPIF DEF DEV,N-$3088G00,M-2,SDA-32/0230,CL-B

System: MPIF0018I 06.56.47 DEVICE PARAMETERS DEFINED
        DEV-NAME MOD  NL    SDA    STATUS  CLASS
        $3088G00 2    32 0230/024F ONLINE  B
        END OF DISPLAY
```

The path class of the device that was defined in the previous example is changed to B and C in the following example.

```
User:   ZMPIF DEF DEV,N-$3088G00,CL-BC

System: MPIF0018I 06.56.47 DEVICE PARAMETERS DEFINED
        DEV-NAME MOD  NL    SDA    STATUS  CLASS
        $3088G00 2    32 0230/024F ONLINE  BC
        END OF DISPLAY
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF DEFINE PATH–Define a MPIF Path

Use this command to do the following:

- Define a new MPIF path
- Change an existing MPIF path definition
- Display information about the MPIF paths that are defined for the next initial program load (IPL).

Requirements and Restrictions

- New and changed path definitions do not take effect until the next IPL.
- You can define or change path definitions only after MPIF restart is completed. Before and during MPIF restart, you can use this command only to display information about the definitions.
- Enter the ZMPIF SET CLASS command before you define any MPIF paths for that class.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - If the command is entered on a processor that does not include 32-way loosely coupled processor support, any definitions that are entered must be reentered when 32-way loosely coupled processor support is installed on the processor.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any definitions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor. If you restore 32-way loosely coupled processor support after falling back to the previous level, the new definitions are also restored.

Format

```
►►──ZMPIF DEFine Path──┬──────────────────────────────────┬──────────►◄
                       │     ┌─,────────────────────┐     │
                       │     ▼                      │     │
                       └─,───┬─Name-path──────────┬─┴─────┘
                             ├─System-sysname─────┤
                             ├─┬─Device-CTOC─┬────┤
                             │ └─Device-MTOM─┘    │
                             ├─┬─Class-A────┬─────┤
                             │ └─Class-code─┘     │
                             └─┬─STatus-START─┬───┘
                               └─STatus-STOP──┘
```

Name-*path*
is a 1- to 8-character alphanumeric path name.

Note: You must specify this parameter when you define a new path or change an existing path definition. The path name that you specify cannot already be defined as a device name.

System-*sysname*
is the 1- to 8-character name of the system at the other end of the path.

Note: You must specify this parameter when you define a new path.

Device
specifies the path type, where:

CTOC
defines a channel-to-channel path.

MTOM
defines a memory-to-memory path.

Class-*code*
is a 1-character alphanumeric code that identifies the path class for the device.

STatus
specifies the IPL status of the path, where:

START
allows physical links to be assigned immediately when you perform an IPL.

STOP
does not allow physical links to be assigned when you perform an IPL until you enter the ZMPIF START command for the path.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF DEFINE HELP**.
- You can define more than one path to a system as long as you use unique path names for each path.
- You can specify the parameters for this command in any order.
- To define a new path, you must specify both the NAME and SYSTEM parameters.
- To change an existing path definition, you must specify the NAME parameter and any other parameters that you want to change.
- Enter **ZMPIF DEFINE PATH** (with no parameters) to display the paths that are defined for the next IPL.
- To display the path definition for a specific path, specify only the NAME parameter.
- To change the name of a path, enter the ZMPIF DELETE command to delete the path and then enter the ZMPIF DEFINE PATH command to redefine the path.
- To display information about the paths that are currently defined, enter **ZMPIF DISPLAY PATH,ALL**.

Examples

The following information is displayed in the examples:

PATHNAME
is the name of the path.

DEVICE
is the device type, which can be:

CTOC
: Channel-to-channel.

MTOM
: Memory-to-memory.

SYSTEM
: is the name of the system to which the path is connected.

CL
: is the path class.

STATUS
: is the status of the path, which can be:

START
: Physical links can be assigned immediately when an IPL is performed.

STOP
: Physical links cannot be assigned when an IPL is performed until the ZMPIF START command is entered for the path.

The following example displays information about the paths that are defined for the next IPL.

```
User:   ZMPIF DEF PATH

System: MPIF0001I 06.56.47 PATH PARAMETERS DEFINED
        PATHNAME DEVICE SYSTEM   CL STATUS
        CPUB.A1   MTOM  CPUB     A  START
        CPUB.B1   MTOM  CPUB     B  START
        CPUC.A1   CTOC  CPUC     A  START
        CPUC.B1   CTOC  CPUC     B  START
        CPUC.C1   CTOC  CPUC     C  START
        END OF DISPLAY
```

A new path is defined in the following example.

```
User:   ZMPIF DEF PATH,N-CTC001,SYS-TPF01,DEV-CTOC,CL-B,STATUS-START

System: MPIF0001I 06.56.47 PATH PARAMETERS DEFINED
        PATHNAME DEVICE SYSTEM   CL STATUS
        CTC001    CTOC  TPF01    B  START
        END OF DISPLAY
```

The code for the path class of the device that was defined in the previous example is changed from B to C in the following example.

```
User:   ZMPIF DEF PATH,N-CTC001,CL-C

System: MPIF0001I 06.56.47 PATH PARAMETERS DEFINED
        PATHNAME DEVICE SYSTEM   CL STATUS
        CTC001    CTOC  TPF01    C  START
        END OF DISPLAY
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF DELETE–Delete a Path or Device

Use this command to delete a path or device definition.

Requirements and Restrictions

- Enter the ZMPIF STOP command to stop the device or path before you delete it.
- If your loosely coupled complex includes processors that are using a TPF image with 32-way loosely coupled processor support and other processors that are using images that do not include 32-way loosely coupled processor support, be aware of the following:
  - If the command is entered on a processor that does not include 32-way loosely coupled processor support, any deletions that are entered must be reentered when 32-way loosely coupled processor support is installed on the processor.
  - If the command is entered on a processor that includes 32-way loosely coupled processor support, any deletions that are entered will be lost and must be reentered if you fall back to a previous level of support on that processor. If you restore 32-way loosely coupled processor support after falling back to the previous level, the deletions are also restored.

Format

```
►►──ZMPIF DELete── ──┬─Path,Name-path─────┬──────────────────────────────────►◄
                     └─Device,Name-$device─┘
```

Path
deletes a path definition.

Name-*path*
is a 1- to 8-character alphanumeric path name.

Device
deletes a device definition.

Name-$*device*
is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

Additional Information

Online help information is available for this command. To display the help information, enter **ZMPIF DELETE HELP**.

Examples

The specified path definition is deleted in the following example.

```
User:   ZMPIF DELETE PATH,N-CTC001

System: MPIF0069I 06.56.47 PATH DELETED
```

The specified device definition is deleted in the following example.

```
User:   ZMPIF DELETE DEVICE,N-$3088G00

System: MPIF0069I 06.56.47 DEVICE DELETED
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF DISPLAY–Display Parameters

Use this command to display formatted information about the status and activity of MPIF resources.

Requirements and Restrictions

This command displays information about the MPIF resources that were defined before you performed the last initial program (IPL). If you defined any new resources or changed any existing resources since that time, the new or changed information is not displayed until after you perform another IPL.

Format

```
►►──ZMPIF DISPlay── ──┬─Activity,─┬─Path-path──────┬─┬────────────────►◄
                      │           ├─Device-$device─┤ │
                      │           ├─Class-code─────┤ │
                      │           └─ALL────────────┘ │
                      ├─CLass,─┬─Code-code─┬─────────┤
                      │        └─ALL───────┘         │
                      ├─COMplex──────────────────────┤
                      ├─Connect,─┬─User-user──────┬──┤
                      │          ├─Device-$device─┤  │
                      │          ├─Class-code─────┤  │
                      │          └─ALL────────────┘  │
                      ├─Device───────────────────────┤
                      ├─Path,─┬─Name-path──────┬─────┤
                      │       ├─Device-$device─┤     │
                      │       ├─Class-code─────┤     │
                      │       └─ALL────────────┘     │
                      ├─Status───────────────────────┤
                      └─SYstem───────────────────────┘
```

Activity
displays information about the activity of the MPIF resources.

Path-*path*
displays information about the activity on a specific path, where *path* is a 1- to 8-character alphanumeric path name.

Device-$*device*
displays information about the path activity through a specific device, where $*device* is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

Class-*code*
displays information about the activity of the paths that are assigned to a specific path class, where *code* is a 1-character alphanumeric path class code.

ALL
displays information about all the path activity in the system.

CLass
displays information about path classes.

Code-*code*
displays information about a specific path class, where *code* is a 1-character alphanumeric path class code.

ALL
displays information about all the path classes.

COMplex
displays the following information about the complex:
- Maximum number of systems allowed in the complex
- Maximum number of users allowed in the complex
- Timeout value for establishing connections
- Maximum time, in seconds, for an I/O event to end
- Timeout value for stalled paths.

Connect
displays information about the connections defined for the MPIF resources.

User-*user*
displays information about the connections defined for a specific MPIF user, where *user* is a 1- to 8-character alphanumeric user name.

Device-$*device*
displays information about the connections through a specific device, where $*device* is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

Class-*code*
displays information about the connections that are assigned to a specific path class, where *code* is a 1-character alphanumeric path class code.

ALL
displays information about all the MPIF connections.

Device
displays the information that is defined in the MPIF hardware table (HDW), which is contained in the DCTCWA DSECT.

Path
displays information about the paths that are defined for the MPIF resources.

Name-*path*
displays information about a specific MPIF path, where *path* is a 1- to 8-character alphanumeric path name.

Device-$*device*
displays information about the paths to a specific device, where $*device* is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

Class-*code*
displays information about the paths that are assigned to a specific path class, where *code* is a 1-character alphanumeric path class code.

ALL
displays information about all the MPIF paths.

Status
displays the following status information about the MPIF resources:
- System name
- Interface version number
- Trace activity.

SYstem
displays the following information about the MPIF system:
- Name of the system
- Number of active paths
- Maximum number of connections allowed in the system
- Maximum number users allowed in the system
- Number of path notification activation entries
- Number of directory notification entries
- Maximum queue depth.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF DISPLAY HELP**.
- Enter the ZMPIF DEFINE or ZMPIF SET command to display information about the resources that are defined for the next IPL.

Examples

Information about the activity on the specified path is displayed in the following example, where:

DEV-NAME
is the name of the device.

PATHNAME
is the name of the path.

CL
is the path class.

M-RATE
is the message rate per second.

MSGSIZE
is the average number of bytes per message.

READS
is the number of read operations per second.

WRITES
is the number of write operations per second.

QUEUED
is the number of messages queued per second.

OV-RUN
is the number of queue overruns per second.

```
User:   ZMPIF DISP A,P-CPUC.A1

System: MPIF0017I 06.56.47 ACTIVITY INFORMATION
        DEV-NAME PATHNAME CL M-RATE MSGSIZE READS  WRITES QUEUED OV-RUN
        $CTOC03  CPUC.A1  A  0004.5    2099 0000.0 0001.0 004.50 0000.0
        END OF DISPLAY
```

Information about the C path class is displayed in the following example, where:

CL
is the path class.

RBUFF
is the read buffer size.

WBUFF
is the write buffer size.

BLOCK
specifies whether or not requests that are unrelated to the path class are blocked.

LOADB
specifies whether load balancing is performed or not.

SHARED
specifies whether the path can be shared or not.

PROTECT
specifies whether the last path in the path class is protected from being stopped or not when you enter the ZMPIF STOP command.

```
User:   ZMPIF DISP CL,CO-C

System: MPIF0032I 06.56.47 ACTIVE CLASS PARAMETERS
        CL RBUFF WBUFF BLOCK LOADB SHARED PROTECT
        C  30720 30720 YES   YES   YES    NO
        END OF DISPLAY
```

Information about the complex is displayed in the following example, where:

NSYS
is the maximum number of systems allowed in the complex.

NUSER
is the maximum number of users allowed in the complex.

CONTIME
is the timeout value, in seconds, for establishing connections.

PATHTIME
is the maximum time, in seconds, for an I/O event to end.

SYSTEM
is the timeout value, in seconds, for stalled paths.

```
User:   ZMPIF DISP COMP

System: MPIF0031I 06.56.47 ACTIVE COMPLEX PARAMETERS
        NSYS- 8 NUSER-   16 CONTIME-80 PATHTIME-81 SYSTIME-  82
```

Information about all the MPIF connections is displayed in the following example, where:

MPIFUSER
is the name of the user in the current system.

STATUS
is the user status, which can be:

ACTIVE
User is active.

INACTIVE
User is not active.

DISC-REQ
User is disconnecting.

CONN-REQ
User is establishing a connection.

PATHNAME
is the name of the path.

CL
is the path class.

DEV-NAME
is the name of the device.

SYSTEM
is the name of the connected system.

CONUSER
is the name of the user in the connected system.

```
User:   ZMPIF DISP C,ALL

System: MPIF0014I 06.56.47 ACTIVE CONNECTION PARAMETERS
        MPIFUSER STATUS   PATHNAME CL DEV-NAME SYSTEM   CONUSER
        MPIF     ACTIVE   CPUB.A1  A           CPUB     MPIF
        MPIF     ACTIVE   CPUB.B1  B           CPUB     MPIF
        MPIF     ACTIVE   CPUC.A1  A  $CTOC03  CPUC     MPIF
        IPC      ACTIVE   CPUC.A1  A  $CTOC03  CPUC     IPC
        END OF DISPLAY
```

Information about all the MPIF devices is displayed in the following example, where:

DEV-NAME
is the name of the device.

MOD
is the model number of the device

NL
is the number of physical links defined for the device.

SDA
is the symbolic device address range for the device.

STATUS
is the status of the device, which can be:

ACTIVE
Device is started.

INACTIVE
Device is not started.

CLASS
is the path classes assigned to the device.

```
User:   ZMPIF DISP DEV

System: MPIF0019I 06.56.47 ACTIVE DEVICE PARAMETERS
        DEV-NAME MOD  NL    SDA    STATUS   CLASS
        $CTOC01  C    16 0900/090F ACTIVE   ABC
        $CTOC02  C    16 0A00/0A0F ACTIVE   ABC
        $CTOC03  2     4 0388/038B ACTIVE   ABC
        $CTOC04  2     4 03A8/03AB ACTIVE   ABC
        $CTOC05  2     8 1388/138F ACTIVE   ABC
        $CTOC06  2    20 13A8/13BB ACTIVE   ABC
        $CTOC07  2     4 238C/238F ACTIVE   ABC
        $CTOC08  2     4 23B8/23BB ACTIVE   ABC
        $CTOC09  2    20 33A4/33B7 ACTIVE   ABC
        $CTOC10  2    20 43A0/43B3 ACTIVE   ABC
        $CTOC11  2    16 53A0/53AF ACTIVE   ABC
        END OF DISPLAY
```

Information about the paths that are assigned to the A path class is displayed in the following example, where:

PATHNAME
: is the name of the path.

SYSTEM
: is the name of the system to which the path is connected.

DEVICE
: is the device type, which can be:

  CTOC
  : Channel-to-channel.

  MTOM
  : Memory-to-memory.

CL
: is the path class.

CNT
: is the number of connections on the path.

STATUS
: is the status of the path, which can be:

  ACTIVE
  : Path is active.

  INACTIVE
  : Path is not active.

  START-UP
  : Path is being started.

  STOPPED
  : Path is being stopped.

DEV-NAME
: is the name of the device associated with the path.

SDA
: is the symbolic device addresses (SDAs) of the read and write devices allocated to the path.

```
User:   ZMPIF DISP P,CL-A

System: MPIF0015I 06.56.47 ACTIVE PATH PARAMETERS
        PATHNAME SYSTEM  DEVICE CL CNT STATUS   DEV-NAME    SDA
        CPUB.A1  CPUB     MTOM  A  000 ACTIVE             0000/0000
        CPUC.A1  CPUC     CTOC  A  001 ACTIVE   $CTOC03  0388/0389
        CPUD.A1  CPUD     CTOC  A  000 INACTIVE           0000/0000
        CPUE.A1  CPUE     CTOC  A  000 INACTIVE           0000/0000
        CPUZ.A1  CPUZ     CTOC  A  000 INACTIVE           0000/0000
        CPU0.A1  CPU0     CTOC  A  000 INACTIVE           0000/0000
        CPU1.A1  CPU1     CTOC  A  000 INACTIVE           0000/0000
        CPU2.A1  CPU2     CTOC  A  000 INACTIVE           0000/0000
        END OF DISPLAY
```

Status information about MPIF is displayed in the following example, where:

SYSTEM NAME
: is the name of the system.

VERSION
: is the version number of the cross-system interface.

TRACE
: specifies whether the trace facility is on or off.

```
User:   ZMPIF DISP STAT

System: MPIF0016I 06.56.47 STATUS INFORMATION
        SYSTEM NAME-CPUB        VERSION-   3
        TRACE-OFF
```

Information about the MPIF system is displayed in the following example, where:

NAME
: is the name of the system.

NPATH
: is the number of active paths.

NCONN
: is the maximum number of connections allowed in the system.

NUSER
: is maximum number of users allowed in the system.

NPAN
: is the number of path notification activation entries.

NDNT
: is the number of directory notification entries.

QDEPTH
: is the maximum queue depth.

```
User:   ZMPIF DISP SYS

System: MPIF0030I 06.56.47 ACTIVE SYSTEM PARAMETERS
        NAME-CPUB     NPATH-  23 NCONN-  11 NUSER-   8
        NPAN- 90      NDNT- 90   QDEPTH- 10
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF HDW–Hardware Table Record Verification

Use this command to initialize the MPIF hardware table records.

Requirements and Restrictions

- You can enter this command only when you are prompted by the TPF system.
- You must perform an initial program load (IPL) for the TPF system after you initialize the hardware table records.

Format

```
►►──ZMPIF HDW── ──┬─INIT───┬──────────────────────────────────────────────►◄
                  └─BYPASS─┘
```

INIT
: initializes the hardware table records with the correct record identification and record code check bytes. The remainder of each data record is set to 0. The MPIF area in keypoint E (CTKE) is also initialized.

BYPASS
: does not initialize the hardware table records and ends MPIF restart.

Additional Information

Online help information is available for this command. To display the help information, enter **ZMPIF HDW HELP**.

Examples

The hardware table records are initialized in the following example.

```
System: CBR10001E 16.12.01
        MPIF DATA IN HARDWARE TABLE CANNOT BE ACCESSED DUE TO RECORD ID ERROR
        SPECIFY OPTION:
        TO INITIALIZE HARDWARE TABLE ENTER:
         ZMPIF HDW INIT
        TO BYPASS THE INITIALIZATION AND ABORT MPIF RESTART:
         ZMPIF HDW BYPASS

User:   ZMPIF HDW INIT

System: CSMP0099I 16.12.01 010000-C ZMPIF HDW INIT
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF KPE–Keypoint E Verification

Use this command to replace the MPIF data in keypoint E (CTKE) with valid data.

Requirements and Restrictions

- You can enter this command only when you are prompted by the TPF system.
- When you replace the data in keypoint E, the TPF system continues to cycle to 1052 state. However, you must perform an initial program load (IPL) to rebuild the MPIF tables using the new data.
- If you do not replace the data in keypoint E, enter the ZMPIF SET and ZMPIF DISPLAY commands to verify the integrity of the MPIF database.

Format

```
►►──ZMPIF KPE── ──┬─REP────┬──────────────────────────────────────────►◄
                  └─BYPASS─┘
```

REP
replaces the MPIF data in keypoint E with valid data from the path definition record ordinal 1 associated with the current system.

Note: A dump is performed when you specify this parameter.

BYPASS
does not replace the data in keypoint E. The TPF system continues to cycle to 1052 state.

Note: When you specify this parameter, the MPIF tables may not be complete or usable. This can cause problems when other MPIF programs use the data in the MPIF tables.

Additional Information

Online help information is available for this command. To display the help information, enter **ZMPIF KPE HELP**.

Examples

The MPIF data in keypoint E is replaced in the following example.

```
System: CBR20006A 16.12.01
        MPIF DATA IN KEYPOINT E CANNOT BE VALIDATED
        SPECIFY OPTION:
        TO REPLACE MPIF DATA IN CTKE - ENTER :
           ZMPIF KPE REP
              IPL TO REBUILD MPIF CONTROL TABLES
        TO BYPASS THE REPLACE:
           ZMPIF KPE BYPASS

User:   ZMPIF KPE REP

System: CSMP0099I 16.12.01 010000-C ZMPIF KPE REP
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF PDR–Path Definition Record Verification

Use this command to initialize MPIF path definition records (PDRs).

Requirements and Restrictions

- You can enter this command only when you are prompted by the TPF system.
- You must perform an initial program load (IPL) for the TPF system after you initialize the path definition records.

Format

```
►►──ZMPIF PDR── ──┬─INIT─┬──────┬─┬──────────────────────────────────►◄
                  │      └──ALL─┘ │
                  └─BYPASS────────┘
```

INIT
initializes the processor unique path definition records (#PDREU) for this processor with the correct record identification and record code check bytes. The remainder of each data record is set to 0. You will be prompted to replace data in keypoint E by using the ZMPIF KPE command.

ALL
initializes all PDRs for all processors. If you do not specify this parameter, only records that are not initialized for the processor on which the command is entered are initialized.

BYPASS
does not initialize the path definition records and ends MPIF restart.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF PDR HELP**.
- If you enter **ZMPIF PDR INIT**, the command only initializes records that are not initialized for the processor on which it is entered. To initialize all PDRs on all processors, enter **ZMPIF PDR INIT ALL**.

Examples

The path definition records are initialized in the following example.

```
System: CBR10001E 16.12.01
        MPIF DATA IN PATH DEFINITION RECORDS CANNOT BE ACCESSED DUE TO RECORD ID
        SPECIFY OPTION:
        TO INITIALIZE PATH DEFINITION RECORDS ENTER:
         ZMPIF PDR INIT
        TO BYPASS THE INITIALIZATION AND ABORT MPIF RESTART:
         ZMPIF PDR BYPASS

User:   ZMPIF PDR INIT

System: CSMP0099I 16.12.01 010000-C ZMPIF PDR INIT
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF SET–Set MPIF Parameters

Use this command to do the following :

- Define a new path class
- Change an existing system, complex, or path class definition
- Display information about the MPIF system, complex, or path classes that are defined for the next initial program load (IPL).

Requirements and Restrictions

- New and changed path class, complex, and system definitions do not take effect until the next IPL.
- If you change a path class definition, the existing path definitions that are assigned to that path class are ***not*** updated automatically. You must do the following to redefine the path definition:
  1. Enter the ZMPIF STOP command to stop the path.
  2. Enter the ZMPIF DELETE command to delete the path definition.
  3. Enter the ZMPIF DEFINE PATH command to redefine the path.
- You can define or change path class, complex, and system definitions only after MPIF restart is completed successfully. Before and during MPIF restart, you can use this command only to display information about the definitions.

Format

```
>>--ZMPIF SET-- --+-CLass--+-------------------------------+----+------------><
                  |        |   .-,------------------.      |    |
                  |        |   V                    |      |    |
                  |        '-,---+-Code-code------+-+------'    |
                  |              +-Rbuff-size-----+             |
                  |              +-Wbuff-size-----+             |
                  |              |        .-NO--. |             |
                  |              +-Block--+-----+-+             |
                  |              |        '-YES-' |             |
                  |              |        .-NO--. |             |
                  |              +-Loadb--+-----+-+             |
                  |              |        '-YES-' |             |
                  |              |         .-YES-.|             |
                  |              +-Shared--+-----++             |
                  |              |         '-NO--'|             |
                  |              |          .-NO--.             |
                  |              '-Protect--+-----+             |
                  |                         '-YES-'             |
                  +-Complex--+-------------------------------+--+
                  |          |   .-,------------------.      |  |
                  |          |   V                    |      |  |
                  |          '-,---+-Nsys-sysmax-----+-+-----'  |
                  |                +-NUser-usrmax----+          |
                  |                +-Contime-time----+          |
                  |                +-Pathtime-time---+          |
                  |                '-Systime-time----'          |
                  '-System--+-------------------------------+---'
                            |   .-,------------------.      |
                            |   V                    |      |
                            '-,---+-Name-sysname----+-+-----'
                                  +-NConn-conmax----+
                                  +-NUser-usrmax----+
                                  +-NPan-entries----+
                                  +-NDnt-entries----+
                                  '-Qdepth-depth----'
```

CLass
defines a new path class, changes an existing path class definition, or displays the path class definitions for the next IPL.

Code-*code*
is a 1-character alphanumeric path class code that identifies the path class.

Rbuff-*size*
is the maximum size of the read staging buffer, where *size* is a 1- to 5-digit decimal number. The maximum value is 63 488.

Wbuff-*size*
is the maximum size of the write staging buffer, where *size* is a 1- to 5-digit decimal number. The maximum value is 63 488.

Block
specifies whether requests that are unrelated to the path class are processed or not.

Note: You cannot override this parameter during cross-system startup processing to accommodate connecting to a system that cannot support blocking.

Loadb
specifies whether load balancing is performed for the path class or not.

Shared
specifies whether the path class can be shared by different MPIF users or not. If the path class is dedicated to one user, only one connection on each path in the path class is allowed.

Protect
specifies whether the last path in the path class is protected from being stopped or not when you enter the ZMPIF STOP command.

Complex
changes the existing complex definition or displays the complex definition for the next IPL.

Nsys-*sysmax*
is the maximum number of systems allowed in the complex, where *sysmax* is a 1- to 2-digit decimal number.

NUser-*usrmax*
is the maximum number of users allowed in the complex, where *usrmax* is a 1- to 5-digit decimal number.

Contime-*time*
is the timeout value, in seconds, for establishing connections, where *time* is a 1- to 2-digit decimal number.

Note: The suggested value for this parameter is 30 seconds.

Pathtime-*time*
is the maximum time, in seconds, for an I/O event to end, where *time* is a 1- to 2-digit decimal number. This value is used to detect stalled paths.

Note: The suggested value for this parameter is 3 seconds.

Systime-*time*
is the timeout value, in seconds, for stalled paths, where *time* is a 1- to 4-digit decimal number. This value should accommodate the time required to recover from a low core condition or to perform a recoverable system error dump, whichever is greater.

Note: The suggested value for this parameter is 15 seconds.

SYstem
changes the existing system definition or displays the system definition for the next IPL.

Name-*sysname*
is the 1- to 8-character alphanumeric system name; for example, TPF.

NConn-*conmax*
is the maximum number of connections allowed in the system, where *conmax* is a 1- to 4-digit decimal number.

NUser-*usrmax*
is the maximum number of users allowed in the system, where *usrmax* is a 1- to 4-digit decimal number.

NPan-*entries*
is the number of path notification activation entries for the system, where *entries* is a 1- to 3-digit decimal number.

NDnt-*entries*
is the number of directory notification entries for the system, where *entries* is a 1- to 3-digit decimal number.

Qdepth-*depth*
is the send queue depth that determines when MPIF stops sending messages, where *depth* is a 1- to 3-digit decimal number.

Note: The suggested value for this parameter is 6 messages.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF SET HELP**.
- When you specify a value for the RBUFF and WBUFF parameters, consider the following:
  - If the value is not on a 2 KB boundary, MPIF automatically rounds up the value to the nearest 2 KB boundary.
  - The maximum valid value for these parameters are 63 488.
  - For IPC, the minimum valid value for these parameters is 4096.
  - If MPIF receives only single data records, the value of these parameters indicates the actual number of data records that are received. In this case, the maximum data size must fit in one of the standard TPF core block sizes; for example, the 1055-byte core block.
  - If blocking is used, the maximum number of data characters is received in blocked format. You must account for the multisystem request block (MSRB) size when determining the value for these parameters because an MSRB is required for each distinct data message blocked to a MPIF staging buffer.
  - If you use the list format, the value of these parameters must include the size of a standard TPF core block that is large enough to contain the list that is provided.
- You can specify the parameters for this command in any order.
- To define a new path class, you must specify the CODE, RBUFF, and WBUFF parameters.
- To change an existing path class definition, you must specify the CODE parameter and any other parameters that you want to change.
- Enter **ZMPIF SET CLASS** (with no parameters) to display the path classes that are defined for the next IPL.
- Enter **ZMPIF SET COMPLEX** (with no parameters) to display the complex that is defined for the next IPL.
- Enter **ZMPIF SET SYSTEM** (with no parameters) to display the system that is defined for the next IPL.
- To display the class definition for a specific path definition, specify only the CODE parameter.
- To display information about the path classes that are currently defined, enter **ZMPIF DISPLAY CLASS,ALL**.
- To display information about the complex that is currently defined, enter **ZMPIF DISPLAY COMPLEX**.

- To display information about the system that is currently defined, enter **ZMPIF DISPLAY SYSTEM**.

Examples

The path class definitions for the next IPL are displayed in the following example, where:

CL
: is the path class.

RBUFF
: is the read buffer size.

WBUFF
: is the write buffer size.

BLOCK
: specifies whether requests that are unrelated to the path class are processed or not.

LOADB
: specifies whether load balancing is performed or not.

SHARED
: specifies whether the path can be shared or not.

PROTECT
: specifies whether the last path in the path class is protected from being stopped or not when you enter the ZMPIF STOP command.

```
User:   ZMPIF SET CL

System: MPIF0006I 06.56.47 CLASS PARAMETERS SET
        CL RBUFF WBUFF BLOCK LOADB SHARED PROTECT
        A  61440 61440 YES   NO    YES    YES
        B   4096  4096 NO    YES   YES    NO
        C  30720 30720 YES   YES   YES    NO
        END OF DISPLAY
```

A new path class is defined in the following example.

```
User:   ZMPIF SET CL,CO-E,RBUFF-4096,WBUFF-4096

System: MPIF0006I 06.56.47 CLASS PARAMETERS SET
        CL RBUFF WBUFF BLOCK LOADB SHARED PROTECT
        E   4096  4096 NO    NO    YES    NO
        END OF DISPLAY
```

The A path class definition is changed in the following example. Blocking is now performed in this path class.

```
User:   ZMPIF SET CL,CO-A,BLOCK-YES

System: MPIF0006I 06.56.47 CLASS PARAMETERS SET
        CL RBUFF WBUFF BLOCK LOADB SHARED PROTECT
        A  61440 61440 YES   NO    YES    YES
        END OF DISPLAY
```

The complex definition for the next IPL is displayed in the following example, where:

NSYS
: is the maximum number of systems that can be connected to the complex.

NUSER
: is the maximum number of users allowed in the complex.

CONTIME
: is the timeout value, in seconds, for establishing a connection.

PATHTIME
: is the maximum time, in seconds, for an I/O event to end.

SYSTIME
: is the timeout value, in seconds, for stalled paths.

```
User:   ZMPIF SET COMP

System: MPIF0005I 06.56.47 COMPLEX PARAMETERS SET
        NSYS-10 NUSER-  100 CONTIME-80 PATHTIME-81 SYSTIME-  82
```

The current complex definition is changed in the following example. The maximum number of users is changed to 500.

```
User:   ZMPIF SET COMP,NUSER-500

System: MPIF0005I 06.56.47 COMPLEX PARAMETERS SET
        NSYS-10 NUSER-  500 CONTIME-80 PATHTIME-81 SYSTIME-  82
```

The system definition for the next IPL is displayed in the following example, where:

NAME
: is the name of the system.

NCONN
: is the maximum number of connections allowed in the system.

NUSER
: is maximum number of users allowed in the system.

NPAN
: is the number of path notification activation entries.

NDNT
: is the number of directory notification entries.

QDEPTH
: is the maximum queue depth.

```
User:    ZMPIF SET SYS

System: MPIF0004I 06.56.47 SYSTEM PARAMETERS SET
        NAME-CPUB     NCONN-  90 NUSER-  24 NPAN- 90 NDNT- 90 QDEPTH- 10
```

The current system definition is changed in the following example. The maximum number of connections is changed to 100.

```
User:   ZMPIF SET SYS,NCONN-100

System: MPIF0004I 06.56.47 SYSTEM PARAMETERS SET
        NAME-CPUB     NCONN- 100 NUSER-  24 NPAN- 90 NDNT- 90 QDEPTH- 10
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF START–Start a Path or Device

Use this command to start a MPIF path or device.

Requirements and Restrictions

None.

Format

```
►►─ZMPIF STArt─ ──┬─Path,─┬─Name-path──────┬─┬──────────────────────────►◄
                  │       ├─System-sysname─┤ │
                  │       └─ALL────────────┘ │
                  └─Device,Name-─┬─$device─┬─┘
                                 └─ALL─────┘
```

Path
starts a path.

Name-*path*
starts the specified path, where *path* is a 1- to 8-character alphanumeric path name.

System-*sysname*
starts all the paths in the specified system, where *sysname* is a 1- to 8-character alphanumeric system name; for example, TPF.

ALL
starts all the inactive paths.

Device
starts a device.

Name-$*device*
starts the specified device, where $*device* is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

ALL
starts all the offline devices.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF START HELP**.
- Paths that are defined with a START status start automatically when you perform an initial program load (IPL).
- Devices that are defined with an ONLINE status start automatically when you perform an IPL.

Examples

The specified path is started in the following example, where:

SYSTEM
is the name of the system.

CLASS
is the path class assigned to the path.

VIA
is the device.

RSDA
is the symbolic device address of the read device assigned to the path.

WSDA
is the symbolic device address of the write device assigned to the path.

```
User:   ZMPIF START P,N-CPUC.B1

System: MPIF0028I 06.56.47 PATH(S) STARTED
        CBT10003I 13.23.52
        MPIF PATH CPUB.B1  ESTABLISHED TO
        SYSTEM - CPUB      CLASS - B
        VIA - $CTOC05   RSDA - 138A  WSDA - 138B
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF STOP–Stop a Path or Device

Use this command to stop a MPIF path or device.

Requirements and Restrictions

None.

Format

```
►►──ZMPIF STOP── ──┬─Path,──┬─────────┬──┬─Name-path──────┬──┬──────────────►◄
                   │        └─PURGE,──┘  ├─System-sysname─┤  │
                   │                     ├─Device-$device─┤  │
                   │                     └─ALL────────────┘  │
                   └─Device,──┬─────────┬──Name-$device──────┘
                              └─PURGE,──┘
```

Path
stops a path.

PURGE
stops a path immediately and disconnects all the associated connections.

Name-*path*
stops the specified path, where *path* is a 1- to 8-character alphanumeric path name.

System-*sysname*
stops all of the paths for the specified system, where *sysname* is a 1- to 8-character alphanumeric system name; for example, TPF.

Device-$*device*
stops all of the paths for the specified device, where $*device* is a 1- to 7-character alphanumeric generic device name.

ALL
stops all of the inactive paths.

Device
stops a device.

PURGE
stops a device immediately and disconnects all the associated connections.

Name-$*device*
stops the specified device, where $*device* is a 1- to 7-character alphanumeric generic device name.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF STOP HELP**.
- A path stops when there is no more traffic for that path. A device stops when there is no more traffic on any of the paths to the device.

Examples

The specified path is stopped in the following example.

```
User:   ZMPIF STOP PATH,N-CPUB.B1

System: MPIF0027I 13.55.32 PATH(S) QUIESCING

        CBL00001I 14.36.16 STOP PATH WITH QUIESCE COMPLETE FOR PATH CPUB.B1
```

The specified device is stopped immediately in the following example.

```
User:   ZMPIF STOP D,PURGE,N-$CTOC02

System: MPIF0026I 13.43.02 DEVICE PURGED
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMPIF TRACE–Trace a Path or User

Use this command to do the following:

- Trace the activity for a MPIF user, path, device, or system.
- Stop the trace.
- Display the current contents of the multisystem request block (MSRB) trace table.
- Display the current contents of the MPIF I/O trace table.

Requirements and Restrictions

None.

Format

```
►►──ZMPIF TRACE── ──┬─USER-username──────────────────────────┬──────────►◄
                    │            ┌─,─────────┐               │
                    │            ▼           │               │
                    ├─PATH-────┬─pathname─┬──┴───────────────┤
                    │          └─$device──┘                  │
                    ├─ALL────────────────────────────────────┤
                    ├─STOP───────────────────────────────────┤
                    ├─Display────────────────────────────────┤
                    ├─NOLOG──────────────────────────────────┤
                    └─IOD──┬───────────────┬──┬───────────┬──┘
                           ├─ sda──────────┤  └─ ──lniata─┘
                           └─ ──$device D──┘
```

USER-*username*
: traces the activity for the specified MPIF user, where *user* is a 1- to 8-character alphanumeric user name.

PATH
: traces a path.

  pathname
  : traces the activity on the specified path, where *path* is a 1- to 8-character alphanumeric path name.

  $*device*
  : traces the activity on all of the paths to the specified device, where $*device* is a 1- to 7-character alphanumeric generic device name that begins with the $ character.

ALL
: traces the activity on all of the paths in the system.

STOP
: stops the trace.

Display
: displays the MSRB trace table.

NOLOG
: does not write the MSRB trace table to the real-time tape. The core block that is used to buffer the information that is being displayed continues to wrap.

IOD
displays the MPIF I/O trace table.

sda
displays the contents of the MPIF I/O trace table for a subchannel pair, where *sda* is the ***even*** 3- to 4-digit hexadecimal symbolic device address (SDA) for the subchannel pair.

$*device* **D**
displays the MPIF I/O trace table for all of the paths to the specified device, where $*device* is a 1- to 7- character alphanumeric generic device name that begins with the $ character and is followed by a D.

lniata
prints the MPIF I/O trace table to the specified printer, where *lniata* is the 6-digit hexadecimal line number, interchange address, and terminal address (LNIATA) of the printer.

Additional Information

- Online help information is available for this command. To display the help information, enter **ZMPIF TRACE HELP**.
- You can trace the activity on as many as 8 paths or devices at one time.

Examples

The activity on all of the paths in the system is traced in the following example.

```
User:   ZMPIF TRACE ALL

System: MPIF0010I 13.40.25 MSRB TRACE STARTED
```

The MSRB trace table is displayed in the following example, where:

IO indicates read or write.

D-TOKEN
is the destination token.

S-TOKEN
is the source token.

PATH
is the path name.

FC
is the function code.

PATH-SEQ
is the path send sequence number.

DATA-LEN
is the size of the data area.

PACING
is the pacing count.

P1
is the prefix flag 1.

P2
is the prefix flag 2.

CON-SEQ
is the connection send sequence number.

```
User:   ZMPIF TRACE DISP

System: MPIF0013I 06.56.35 MSRB TRACE TABLE

        IO D-TOKEN S-TOKEN  PATH     FC PATH-SEQ DATA-LEN PACING   P1 P2 CON-SEQ
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000017D 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000017D 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000017D 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000041F 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000041F 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 0000041F 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 00000FFF 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 00000FFF 00000000 80 00 0000002
        R UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 00000FFF 00000000 80 00 0000003
        W UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000019 0000017D 00000000 80 00 0000002
        W UCTOC1A1 UCTOC1A1 CPUC.A1  00 0000001A 0000041F 00000000 80 00 0000002
        W UCTOC1A1 UCTOC1A1 CPUC.A1  00 00000000 00000FFF 00000000 80 00 0000002
        12 ENTRIES IN THE TABLE
```

The MPIF I/O trace table is displayed in the following example, where:

The heading contains the following:

TOP
is the address of the first entry in the MPIF I/O trace table.

CUR
is the address of the current entry in the MPIF I/O trace table.

BOT
is the address of the last entry in the MPIF I/O trace table.

The first line of an entry contains the following:

CMD
is the type of interrupt, for example, SIO, INT, or HDV.

R/W
indicates read (R) or write (W).

SCH
is the symbolic device address (SDA) of the subchannel.

CC
is the condition code.

ADDR
is the address of the post interrupt routine.

NEXT
is the address of the routine that issued the RIOSC macro.

CCW
is the channel command word (CCW) for start subchannel (SSCH) operations.

CSW
is the channel status word (CSW) for interrupts.

The second line of an entry contains the following:

SID
indicates a sense ID command was issued during an SIO interrupt.

SCB
: indicates a sense command byte command was issued during an SIO interrupt.

SAD
: indicates a sense adaptor state command was issued during an SIO interrupt.

SEM
: indicates a set extended mode command was issued during an SIO interrupt.

CTL
: indicates a control command was issued during an SIO interrupt.

PRP
: indicates a prepare command was issued during an SIO interrupt.

cc is the condition code for an INT interrupt.

CWAIND1
: is indicator byte 1 in the CWA table.

IND2
: is indicator byte 2 in the CWA table.

IND3
: is indicator byte 3 in the CWA table.

IND4
: is indicator byte 4 in the CWA table.

IND5
: is indicator byte 5 in the CWA table.

```
User:   ZMPIF TRACE IOD

System: MPIF IO TRACE TABLE ADDRESSES

        TOP 0693D0  CUR 069FF0  BOT 06A370

        CMD R/W SCH CC                ADDR/NEXT             CCW/CSW
        SIO  W  389  0 0000  NEXT 00000000000ADEAE CCW E42000080055A818
          SID    CWAIND1 01  IND2 00  IND3 00  IND4 10  IND5 40  000000
        INT  W  389  0 0000  NEXT 00000000000ADEAE CSW 002020180C000001
          0C     CWAIND1 01  IND2 00  IND3 00  IND4 10  IND5 40  000000
        HDV  W  389  0 0000  ADDR 00000000800ACA00      000000000000000
                 CWAIND1 01  IND2 00  IND3 00  IND4 00  IND5 40  000000
        SIO  R  388  0 0000  NEXT 0000000000068B28 CCW 11460001008D0909
          SCB    CWAIND1 01  IND2 00  IND3 00  IND4 08  IND5 40  000000
        INT  R  388  0 0000  NEXT 0000000000068B28 CSW 008D09280C000001
          0C     CWAIND1 01  IND2 00  IND3 00  IND4 08  IND5 40  000000
        SIO  R  388  0 0000  NEXT 0000000000068C12 CCW 0720000100000000
          CTL    CWAIND1 08  IND2 00  IND3 00  IND4 08  IND5 40  000000
        INT  R  388  0 0000  NEXT 0000000000068C12 CSW 00068FF808000001
          08     CWAIND1 08  IND2 00  IND3 00  IND4 08   IND5 40 000000
```

Related Information

See *TPF Multi-Processor Interconnect Facility Reference* for more information about MPIF.

ZMQID ALTER–Alter the MQI Channel Directory

Use this command to change an existing Message Queue Interface (MQI) channel directory entry.

Requirements and Restrictions

None.

Format

```
►►──ZMQID ALter── ──┬─CHannel-name─┬──────────────────────────────────────────►
                    └─CHL-name─────┘

►──┬────────────────────────────────────────────────────────────────────┬────►
   ├─ COnnection-──┬─hostname─────────────────────────────────────────┤
   │               ├─ipaddr───────┤                                     │
   │               ├─luname───────┤                                     │
   │               └─netid.luname─┘                                     │
   ├─ COnnection-──┬─hostname─┬─ ──TRPtype-TCP─────────────────────────┤
   │               └─ipaddr───┘                                         │
   └─ COnnection-──┬─luname───────┬─ ──TRPtype-APPC──┬────────────────┬─┘
                   └─netid.luname─┘                  └─ ──TPname-tpname─┘

►──┬──────────────────┬──┬─────────────────────────┬──┬────────────────────────┬──►
   └─ ──QMname-qmname─┘  └─ ──DESCR-description─┘  └─ ──MAXMSGL-integer─┘

►──┬────────────────────┬──┬───────────────────────┬──┬────────────────────────┬──►
   └─ ──MOde-modename─┘  └─ ──RCVData-exitdata─┘  └─ ──RCVExit-progname─┘

►──┬───────────────────────┬──┬────────────────────────┬──────────────────────►
   └─ ──SCYData-exitdata─┘  └─ ──SCYExit-progname─┘

►──┬────────────────────────┬──┬─────────────────────────┬────────────────────►◄
   └─ ──SENDData-exitdata─┘  └─ ──SENDExit-progname─┘
```

CHannel-*name*
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

CHL-*name*
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

COnnection
specifies the connection name, where:

hostname
is a 1- to 256-character host name (alphanumeric characters and periods).

ipaddr
is the host Internet Protocol (IP) address (numeric characters and dotted decimal notation).

If the channel connects to a port number other than well-known port 1414, this port number can be specified in parentheses at the end of *hostname*.

luname
is a 1- to 8-character alphanumeric name of the remote partner LU that can contain either of the three *national* characters: at sign (@), number sign (#), and dollar sign ($). If entered in lowercase, *luname* is translated to uppercase.

netid
is a 1- to 8-character alphanumeric network identifier (ID) that can contain either of the three *national* characters: at sign (@), number sign (#), and dollar sign ($). If entered in lowercase, *netid* is translated to uppercase.

TRPtype
specifies the type of communication protocol, where:

APPC
specifies Advanced Program-to-Program Communications (APPC).

TCP
specifies Transmission Control Protocol (TCP).

Tpname-*tpname*
specifies the transaction program name of the server, where *tpname* is a 1- to 64-character name. The transaction program name can contain only displayable characters and can include lowercase letters. Hexadecimal (nondisplayable) transaction program names are not supported.

QMname-*qmname*
specifies the queue manager name to which an MQI application can request a connection, where *qmname* is a 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A queue manager name cannot begin or end with a period and cannot contain two consecutive periods. If the value is NONE (in uppercase) the queue manager name is set to blanks.

DESCR-*description*
provides descriptive information about the channel, where *description* is a 1- to 64-character description. The description can contain only displayable characters and can include lowercase letters. This description is displayed when you enter a ZMQID DISPLAY command. If there are blanks in the description, enclose the text in single quotation marks (') or slashes (/).

MAXMSGL-*integer*
specifies the maximum message length that can be transmitted on the channel, where *integer* is a number from 1 to 30 000. This is compared with the value defined for the server channel entry, and the actual value used is the lower of the two values.

MOde-*modename*
specifies the LU 6.2 mode name to use for connecting to MQSeries, where *modename* is a 1- to 8-character alphanumeric mode name. If entered in lowercase, *modename* is translated to uppercase.

RCVData-*exitdata*
specifies the data to pass to the receive exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

RCVExit-*progname*
specifies the name of the channel receive exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no receive exit is called. If entered in lowercase, *progname* is translated to uppercase.

SCYData-*exitdata*
specifies the data to pass to the security exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

SCYExit-*progname*
specifies the name of the channel security exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no security exit is called. If entered in lowercase, *progname* is translated to uppercase.

SENDData-*exitdata*
specifies the data to pass to the send exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

SENDExit-*progname*
specifies the name of the channel send exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no send exit is called. If entered in lowercase, *progname* is translated to uppercase.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQID HELP
  ZMQID ?
- The ZMQID ALTER command is the TPF system equivalent of the MQSeries ALTER CHANNEL command. The ZMQID ALTER command changes the attributes of client connection channel (CLNTCONN) type entries defined to the TPF system. A corresponding server connection channel (SVRCONN) type, with the same channel name and attributes, must be defined on the MQI queue manager.
- Alterations take affect immediately and only affect new connection requests. Connections already established are unaffected by a ZMQID ALTER command.

Examples

The mode name and transaction program name for channel OS2CHL1 are changed in the following example.

```
User:   ZMQID ALTER CH-OS2CHL1 MODE-TESTPIPE TP-MQTESTTP

System: NMQI0015I 10:40:12 CHANNEL -OS2CHL1 ENTRY UPDATED
        CHANNEL OS2CHL1  CONNECTION NET1.OS2MQLU1
        TRPTYPE      APPC
        MODE         FASTPIPE
        TPNAME       OS2MQTP
        QMNAME       OS2QMGR1
        DESCR        MQ CLIENT CHANNEL TO QMGR1
        RCVDATA      OS2COMPRESS
        RCVEXIT      USR1
        SCYDATA      ENCRYPT1
        SCYEXIT      USR3
        SENDDATA     OS2COMPRESS
        SENDEXIT     USR2
        MAXMSGL      30000
        ALTERED TO -
        CHANNEL OS2CHL1  CONNECTION NET1.OS2MQLU1
        TRPTYPE      APPC
        MODE         TESTPIPE
        TPNAME       MQTESTTP
        QMNAME       OS2QMGR1
        DESCR        MQ CLIENT CHANNEL TO QMGR1
        RCVDATA      OS2COMPRESS
        RCVEXIT      USR1
        SCYDATA      ENCRYPT1
        SCYEXIT      USR3
        SENDDATA     OS2COMPRESS
        SENDEXIT     USR2
        MAXMSGL      30000
        END OF DISPLAY
```

Related Information

See the *MQSeries Distributed Queue Management Guide* and *MQSeries Command Reference* for more information about MQSeries client and server channels.

ZMQID DEFINE–Define an MQI Channel Directory Entry

Use this command to define a Message Queue Interface (MQI) channel directory entry.

Requirements and Restrictions

- Duplicate channel names are not allowed.
- There are a maximum of 50 channels in the channel directory.

Format

```
►►──ZMQID DEFine── ──┬─CHannel-name─┬──────────────────────────────────────────►
                     └─CHL-name─────┘

►─ ──┬─COnnection-─┬─hostname─┬─ ──TRPtype-TCP──────────────────────┬──────────►
     │             └─ipaddr───┘                                    │
     └─COnnection-─┬─luname───────┬─ ──TRPtype-APPC─┬──────────────┬┘
                   └─netid.luname─┘                 └──TPname-tpname┘

►─┬────────────────┬─┬─────────────────────┬─┬─────────────────────┬────────►
  └──QMname-qmname─┘ └──DESCR-description──┘ └──LIKE-channelname──┘

  ┌──MAXMSGL-30000───┐
►─┼──────────────────┼─┬─────────────────┬─┬────────────────────┬──────────►
  └──MAXMSGL-integer─┘ └──MOde-modename──┘ └──RCVData-exitdata──┘

►─┬──────────────────┬─┬────────────────────┬─┬───────────────────┬────────►
  └──RCVExit-progname┘ └──SCYData-exitdata──┘ └──SCYExit-progname─┘

►─┬─────────────────────┬─┬──────────────────────┬─────────────────────────►◄
  └──SENDData-exitdata──┘ └──SENDExit-progname───┘
```

CHannel-_name_
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods. The name must not be the same as any existing channel name.

CHL-_name_
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods. The name must not be the same as any existing channel name.

COnnection
: specifies the connection name, where:

hostname
is a 1- to 256-character host name (alphanumeric characters and periods).

ipaddr
is the host Internet Protocol (IP) address (numeric characters and dotted decimal notation).

If the channel connects to a port number other than well-known port 1414, this port number can be specified in parentheses at the end of *hostname*.

luname
is a 1- to 8-character alphanumeric name of the remote partner LU that can contain either of the three *national* characters: at sign (@), number sign (#), and dollar sign ($). If entered in lowercase, *luname* is translated to uppercase.

netid
is a 1- to 8-character alphanumeric network identifier that can contain either of the three *national* characters: at sign (@), number sign (#), and dollar sign ($). If entered in lowercase, *netid* is translated to uppercase.

TRPtype
specifies the type of communication protocol, where:

APPC
specifies Advanced Program-to-Program Communications (ACCP).

TCP
specifies Transmission Control Protocol (TCP).

Tpname-*tpname*
specifies the transaction program name of the server, where *tpname* is a 1- to 64-character name. The transaction program name can contain only displayable characters and can include lowercase letters. Hexadecimal (nondisplayable) transaction program names are not supported.

QMname-*qmname*
specifies the queue manager name to which an MQI application can request a connection, where *qmname* is a 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A queue manager name cannot begin or end with a period and cannot contain two consecutive periods. If the value is NONE (in uppercase) or the parameter is not specified, the queue manager name is set to blanks.

DESCR-*description*
provides descriptive information about the channel, where *description* is a 1- to 64-character description. The description can contain only displayable characters and can include lowercase letters. This description is displayed when you enter a ZMQID DISPLAY command. If there are blanks in the description, enclose the text in single quotation marks (') or slashes (/).

LIKE-*channelname*
specifies the name of an existing channel entry to be used as a model, where *channelname* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods. When you enter this parameter, the TPF system uses the values from the model. If you do not specify this parameter, the system default values are used.

MAXMSGL-*integer*
specifies the maximum message length that can be transmitted on the channel, where *integer* is a number from 1 to 30 000. This is compared with the value defined for the server channel entry, and the actual value used is the lower of the two values.

MOde-*modename*
specifies the LU 6.2 mode name to use for connecting to MQSeries, where *modename* is a 1- to 8-character alphanumeric mode name. If entered in lowercase, *modename* is translated to uppercase.

RCVData-*exitdata*
specifies the data to pass to the receive exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

RCVExit-*progname*
specifies the name of the channel receive exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no receive exit is called. If entered in lowercase, *progname* is translated to uppercase.

SCYData-*exitdata*
specifies the data to pass to the security exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

SCYExit-*progname*
specifies the name of the channel security exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no security exit is called. If entered in lowercase, *progname* is translated to uppercase.

SENDData-*exitdata*
specifies the data to pass to the send exit, where *exitdata* is a 1- to 32-character string of data. The string of data can contain only displayable characters and can include lowercase letters. If there are blanks in the string of data, enclose the text in single quotation marks (') or slashes (/).

SENDExit-*progname*
specifies the name of the channel send exit program, where *progname* is a 4-character TPF program name. If the value is NONE or the parameter is not specified, no send exit is called. If entered in lowercase, *progname* is translated to uppercase.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQID HELP
  ZMQID ?
- The ZMQID DEFINE command is the TPF system equivalent of the MQSeries DEFINE CHANNEL command. The ZMQID DEFINE command defines a client connection channel (CLNTCONN) type to the TPF system. A corresponding server connection channel (SVRCONN) type, with the same channel name and attributes, must be defined on the MQI queue manager.

- To accommodate long entries, you can specify just the required parameters, CHANNEL and CONNECTION, in the ZMQID DEFINE command. Use the ZMQID ALTER command to specify the other parameters.

Examples

An MQI channel directory entry is defined in the following example.

```
User:   ZMQID DEFINE CH-AIXCHL1 CO-NET1.AIXLU1 MODE-MODE1 QM-AIXQMGR1 TP-AIXMQTP

System: MQID0011I 10:38:33 MQI CHANNEL ENTRY ADDED
```

Related Information

See the *MQSeries Distributed Queue Management Guide* and *MQSeries Command Reference* for more information about MQSeries client and server channels.

ZMQID DELETE–Delete an MQI Channel Directory Entry

Use this command to delete a Message Queue Interface (MQI) channel directory entry.

Requirements and Restrictions

None.

Format

```
►►──ZMQID DELETE── ──┬─CHannel-name─┬─────────────────────────────────►◄
                     └─CHL-name─────┘
```

CHannel-*name*
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

CHL-*name*
: specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQID HELP
  ZMQID ?
- The ZMQID DELETE command is the TPF system equivalent of the MQSeries DELETE CHANNEL command. The ZMQID DELETE command deletes a client connection channel (CLNTCONN) type defined to the TPF system.

Examples

In the following example the definition for channel AIX1 is removed from the system.

```
User:   ZMQID DELETE CH-AIX1

System: NMQI0016I 10:40:12 CHANNEL AIX1 DELETED
```

Related Information

See the *MQSeries Distributed Queue Management Guide* and *MQSeries Command Reference* for more information about MQSeries client and server channels.

ZMQID DISPLAY–Display the MQI Channel Directory

Use this command to display a Message Queue Interface (MQI) channel directory entry or summary information for a set of entries.

Requirements and Restrictions

None.

Format

```
►►──ZMQID DISplay── ──┬─CHannel-name────┬─────────────────────────────────►◄
                      ├─CHL-name────────┤
                      └─QMname-qmname───┘
```

CHannel-*name*
specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

Note: Use the asterisk (*) as a wildcard character to specify a group of channel names. For example, to display information about all channel names beginning with MVSCHL, enter:

ZMQID DISP CHANNEL-MVSCHL*

CHL-*name*
specifies the channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), period (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

Note: Use the asterisk (*) as a wildcard character to specify a group of channel names. For example, to display information about all channel names beginning with OS2CHL, enter:

ZMQID DISP CHL-OS2CHL*

QMname-*qmname*
specifies the queue manager name to which an MQI application can request a connection, where *qmname* is a 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A queue manager name cannot begin or end with a period and cannot contain two consecutive periods.

Note: Use the asterisk (*) as a wildcard character to specify a group of queue manager names. For example, to display information about all queue manager names beginning with AIXQM, enter:

ZMQID DISP QM-AIXQM*

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQID HELP
  ZMQID ?
- The ZMQID DISPLAY command is the TPF system equivalent of the MQSeries DISPLAY CHANNEL command. The ZMQID DISPLAY command displays client connection channel (CLNTCONN) type entries defined to the TPF system.

Examples

The following information is displayed in the examples.

CHANNEL
: is the name of the MQI client channel.

CONNECTION
: is the network-qualified name of the remote partner LU (the LU name defined on the server), a host name, or Internet Protocol (IP) address.

TRPTYPE
: is the type of communication protocol being used.

MODE
: is the LU 6.2 mode name to use for a connection to the server. If the TRPTYPE parameter is set to TCP, this field is not displayed.

TPNAME
: is the transaction program name on the server. If the TRPTYPE parameter is set to TCP, this field is not displayed.

QMNAME
: is the queue manager name to which the MQI application can request a connection.

DESCR
: is the description of the channel entry.

RCVDATA
: is the data passed to the channel receive exit.

RCVEXIT
: is the name of the channel receive exit program.

SCYDATA
: is the data passed to the channel security exit.

SCYEXIT
: is the name of the channel security exit program.

SENDDATA
: is the data passed to the channel send exit.

SENDEXIT
: is the name of the channel send exit program.

MAXMSGL
: is the maximum message length supported on the client side of the connection.

A single MQI client channel entry is displayed in the following example.

```
User:   ZMQID DIS CH-OS2CHL1

System: MQID0012I 10.48.54 START OF ZMQID DISPLAY
        CHANNEL      OS2CHL1
        CONNECTION   NET1.OS2MQLU1
        TRPTYPE      APPC
        MODE         FASTPIPE
        TPNAME       OS2MQTP
        QMNAME       OS2QMGR1
        DESCR        MQ CLIENT CHANNEL TO QMGR1
        RCVDATA      OS2COMPRESS
        RCVEXIT      USR1
        SCYDATA      ENCRYPT1
        SCYEXIT      USR3
        SENDDATA     OS2COMPRESS
        SENDEXIT     USR2
        MAXMSGL      30000
        END OF DISPLAY
```

Summary information about all the MQI channel entries that have a name beginning with the AIX characters is displayed in the following example.

```
User:   ZMQID DISP CHANNEL AIX-*

System: MQID0013I 13.16.12  START OF ZMQID DISPLAY

        CHANNEL       AIXCHL1
         CONNECTION    NET1.AIXLU01
         QMNAME        AIXQMGR1
        CHANNEL       AIXCHL2
         CONNECTION    AIXLU02
         QMNAME        AIXQMGR1
        CHANNEL       AIXCHL3
         CONNECTION    USANET01.AIXLU03
         QMNAME        AIXQMGR3
        CHANNEL       AIXCHL1
         CONNECTION    NET1.AIXLU01
         QMNAME        AIXQMGR1
        END OF DISPLAY
```

Summary information about all the MQI channel entries that have a queue manager name beginning with the QMGR1 characters is displayed in the following example.

```
User:   ZMQID DISP QMNAME-QMGR1*

System: MQID0014I 13.16.12  START OF ZMQID DISPLAY

        QMNAME       QMGR1
         CONNECTION  NET1.AIXQLU1
         CHANNEL     CHL1
        QMNAME       QMGR10
         CONNECTION  MQSVRLU1
         CHANNEL     CHL10
        QMNAME       QMGR11
         CONNECTION  USANET1.MQMVSLU1
         CHANNEL     CHL11
        QMNAME       QMGR12
         CONNECTION  MQMVSLU1
         CHANNEL     CHL12
        QMNAME       QMGR13
         CONNECTION  MQMVSLU1
         CHANNEL     CHL13
        END OF DISPLAY
```

Summary information about all the MQI channel entries defined in the system is displayed in the following example.

```
User:   ZMQID DISP CH-*

System: MQID0013I 13.16.12  START OF ZMQID DISPLAY

        CHANNEL      chl1
         CONNECTION  1.2.3.4
         QMNAME      TPF.QMGR
        CHANNEL      chl2
         CONNECTION  2.3.4.5
         QMNAME      MVS.QMGR
        CHANNEL      chla12
         CONNECTION  TPFNET.MQLU
         QMNAME      TPF.QMGR01
        END OF DISPLAY
```

Related Information

See the *MQSeries Distributed Queue Management Guide* and *MQSeries Command Reference* for more information about MQSeries client and server channels.

ZMQIT–Display the MQI Trace Table Entries

Use this command to display entries from the Message Queue Interface (MQI) trace table in chronological order.

Requirements and Restrictions

You can enter this command only after Systems Network Architecture (SNA) restart is completed.

Format

```
►►──ZMQIT── ──┬─All─┬──┬────────────┬──┬──────────────────┬──────────►
              └─nbr─┘  └─ ──┬─ERR─┬─┘  └─ ──┬─CHannel-name─┬┘
                            ├─REP─┤         └─CHL-name─────┘
                            └─REQ─┘

►──┬───────────────┬──┬──────────────┬──────────────────────────────►◄
   └─ ──PGM-progname┘  └─ ──ECB-address┘
```

All
: displays all the entries in the MQI trace table.

nbr
: is the number of entries to display in the MQI trace table beginning with the entries at the end of the table.

ERR
: displays only the replies to MQI functions that were returned with an error; that is, the completion code (CompCode) value does not equal 0.

REP
: displays only the replies to MQI functions.

REQ
: displays only the requests for MQI functions.

CHANNEL-*name*
: displays only the MQI trace table entries that were processed for channel name, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A channel name cannot begin or end with a period and cannot contain two consecutive periods.

PGM-*progname*
: displays only the MQI trace table entries that were processed by the application program, where *progname* is the 4-character alphanumeric name of the application program. If entered in lowercase, *progname* is translated to uppercase.

ECB-*address*
: displays only the MQI trace table entries that were processed by the entry control block (ECB) at a specified address, where *address* is the 1- to 8-digit hexadecimal system virtual memory (SVM) address.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQIT HELP
ZMQIT ?

Examples

The following information is displayed in the examples:

O indicates that the MQI function request is outbound to the MQ server.

I indicates that the MQI function request is being returned to the application program.

CHANNEL
is the channel name of the MQ server connection; that is, the MQI channel directory (MQICD) entry currently being used as a transport to the MQ server where the MQI function is processed.

COMPCODE
is the completion code returned to the application program after the MQI function is processed. COMPCODE is displayed only for replies to the application program.

REASON
is returned to the application program after the MQI function is processed. REASON is displayed only for replies to the application program.

FUNC TYPE
is the type of MQI function.

PGM
is the name of the application program that is processing the MQI function.

ECB ADDR
is the (system virtual memory) SVM address of the ECB processing the MQI function.

TIME
is the number of seconds between the time the application program processed the MQI function and the time the application program received the results. The time is displayed only for replies to the application program.

The last four entries in the MQI trace table are displayed in the following example.

```
User:   ZMQIT 4

System: NMQT0001I 09.04.31 MQI TRACE TABLE
        I                                          FUNC         ECB
        O   CHANNEL             COMPCODE REASON    TYPE    PGM  ADDR     TIME
        O   OS2QMGR1CHN1                           MQPUT   MQA1 311000
        I   OS2QMGR1CHN1        00000000 00000000 MQPUT   MQA1 311000   0.054
        O   OS2QMGR2CHN2                           MQPUT1  MQA4 301000
        I   OS2QMGR2CHN2        00000000 00000000 MQPUT1  MQA4 301000   0.153
        END OF DISPLAY
```

The last three replies to MQI functions processed over channel OS2QMGR1CHN1 are displayed in the following example.

```
User:   ZMQIT 3 REP CH-OS2QMGR1CHN1

System: NMQT0001I 09.04.31 MQI TRACE TABLE
        I                                          FUNC         ECB
        O   CHANNEL              COMPCODE REASON   TYPE    PGM  ADDR     TIME
        I   OS2QMGR1CHN1         00000000 00000000 MQOPEN  MQA1 311000   0.104
        I   OS2QMGR1CHN1         00000000 00000000 MQPUT   MQA1 311000   0.054
        I   OS2QMGR2CHN1         00000000 00000000 MQCLOSE MQA4 301000   0.013
        END OF DISPLAY
```

Related Information

See *TPF ACF/SNA Network Generation* for more information about how to define the MQI trace table.

ZMQSC ALT CHL–Alter a TPF MQSeries Channel Definition

Use this command to change a TPF MQSeries channel definition.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- The corresponding channel must be restarted to reflect the new definition after this command is used.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                              ┌ ──CPU-localid──────┐
►►──ZMQSC ALT── ──CHL-name────┼────────────────────┼──────────────────────────────►
                              └ ──CPU-──┬cpuid──┬──┘
                                        └ALL────┘

►──┬───────────────────────────────────────────────────────────────┬──────────►◄
   └ ──CHLTYPE-──┬SDR──┤ Sender Channel Parameters ├────────────────┤
                 ├RCVR──┤ Receiver Channel Parameters ├─────────────┤
                 └SVRCONN──┤ Server Connection Channel Parameters ├─┘
```

Sender Channel Parameters:

```
├──┬─────────────────────┬──┬─────────────────────┬──┬─────────────────┬──►
   └─BATCHSZ-batchsz─┘  └─CONNAME-conname─┘  └─DESCR-descr─┘

►──┬─────────────────────┬──┬─────────────────────┬──┬─────────────────────┬──►
   └─LOCALIP-ipaddress─┘  └─MAXMSGL-maxmsgl─┘  └─MSGDATA-msgdata─┘

►──┬──────────────────┬──┬─────────────────────────┬──►
   └─MSGEXIT-─┬─YES─┬─┘  └─NPMSPEED-─┬─FAST───┬─┘
              └─NO──┘                └─NORMAL─┘

►──┬─────────────────────┬──┬─────────────────────┬──┬──────────────────┬──►
   └─SOCKBUF-bufsize─┘  └─SCYDATA-scydata─┘  └─SCYEXIT-─┬─YES─┬─┘
                                                        └─NO──┘

►──┬─────────────────────┬──┬─────────────────┬──────────────────────────┤
   └─SEQWRAP-seqwrap─┘  └─XMITQ-xmitq─┘
```

Receiver Channel Parameters:

```
├──┬─────────────────────┬──┬─────────────────┬──┬─────────────────────┬──►
   └─BATCHSZ-batchsz─┘  └─DESCR-descr─┘  └─MAXMSGL-maxmsgl─┘

►──┬─────────────────────┬──┬─────────────────┬──┬─────────────────────┬──►
   └─MRDATA-mrdata─┘  └─MREXIT-─┬─YES─┬─┘  └─MSGDATA-msgdata─┘
                                └─NO──┘

►──┬──────────────────┬──┬─────────────────────────┬──►
   └─MSGEXIT-─┬─YES─┬─┘  └─NPMSPEED-─┬─FAST───┬─┘
              └─NO──┘                └─NORMAL─┘

►──┬─────────────────────┬──┬─────────────────────┬──┬──────────────────┬──►
   └─SOCKBUF-bufsize─┘  └─SCYDATA-scydata─┘  └─SCYEXIT-─┬─YES─┬─┘
                                                        └─NO──┘

►──┬─────────────────────┬──────────────────────────────────────────┤
   └─SEQWRAP-seqwrap─┘
```

Server Connection Channel Parameters:

```
├──┬─────────────────┬──┬─────────────────────┬──┬───────────────────────┬──────►
   └──DESCR-descr────┘  └──MAXMSGL-maxmsgl────┘  └──SCYDATA-scydata──────┘

►──┬─────────────────────┬──┬─────────────────────┬─────────────────────────┤
   └──SCYEXIT-──┬─YES─┬──┘  └──SOCKBUF-bufsize────┘
                └─NO──┘
```

CHL-*name*
: specifies the name of the channel definition to be changed, where *name* is a 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. This name cannot be changed.

CPU
: changes the message channel definition for the specified central processing unit (CPU), where:

  cpuid
  : is a 1-character CPU identifier (ID).

  ALL
  : changes all the processors in the system.

  localid
  : is the local CPU ID from which the command is currently running. This is the default.

CHLTYPE
: specifies one of the following channel types:

  SDR
  : specifies the sender channel.

  RCVR
  : specifies the receiver channel.

  SVRCONN
  : specifies the server connection channel.

BATCHSZ-*batchsz*
: changes the maximum number of messages that can be sent over a channel before TPF MQSeries guarantees delivery of the messages in the batch. The TPF batch size range is from 1 to 9999 messages. The maximum batch size actually used is the lowest of the following:

  - The batch size of the sending channel
  - The batch size of the receiving channel.

CONNAME-*conname*
: changes the connection name, where *conname* is from 1 to 256 characters. This is either the host name (alphanumeric characters and periods) or the host Internet Protocol (IP) address (numeric characters and dotted decimal notation).

  If the channel connects to a port number other than well-known port 1414, this port number can be specified in parentheses at the end of *conname*, and *conname* must be enclosed in single quotation marks.

This parameter is valid only for sender channels.

DESCR-*descr*
specifies a description of the channel, where *descr* is from 1 to 64 alphanumeric characters. If *descr* is enclosed in single quotation marks, the characters can be mixed case.

LOCALIP-*ipaddress*
specifies which local IP address is used when establishing a connection, where *ipaddress* is a numeric IP address.

MAXMSGL-*maxmsgl*
changes the maximum message length that can be transmitted on the channel, where *maxmsgl* is a number from 0 to the maximum message length specified in the TPF MQSeries profile.

MRDATA-*mrdata*
specifies the data that will be passed to the channel message retry user exit in segment CUIT, where *mrdata* is from 1 to 32 alphanumeric characters. If *mrdata* is enclosed in single quotation marks, the characters can be mixed case.

MREXIT
specifies one of the following:

YES
specifies that the channel message retry user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel message retry user exit.

NO
specifies that the channel message retry user exit in segment CUIT will not be called.

MSGDATA-*msgdata*
specifies the data that will be passed to the channel message user exit in segment CUIT, where *msgdata* is from 1 to 32 alphanumeric characters. If *msgdata* is enclosed in single quotation marks, the characters can be mixed case.

MSGEXIT
specifies one of the following:

YES
specifies that the channel message user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel message user exit.

NO
specifies that the channel message user exit in segment CUIT will not be called.

NPMSPEED
changes the channel speed, where:

FAST
specifies the following:

- For sender channels, both persistent and nonpersistent messages are sent. Delivery is guaranteed for persistent messages only.
- For receiver channels, both nonpersistent messages and persistent messages are received. Receipt by TPF MQSeries is guaranteed only for persistent messages. Nonpersistent messages received on fast receiver channels are given to the TPF MQSeries ROUTC bridge for processing.

NORMAL
specifies the following:

- For sender channels, both persistent and nonpersistent messages are processed by normal sender channels. Delivery is guaranteed for both persistent and nonpersistent messages.
- For receiver channels, receipt by TPF MQSeries for all messages (persistent and nonpersistent) is guaranteed.

SCYDATA-*scydata*
specifies the data that will be passed to the channel security user exit in segment CUIT, where *scydata* is from 1 to 32 alphanumeric characters. If *scydata* is enclosed in single quotation marks, the characters can be mixed case.

SCYEXIT
specifies one of the following:

YES
specifies that the channel security user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel security user exit.

NO
specifies that the channel security user exit in segment CUIT will not be called.

SEQWRAP-*seqwrap*
specifies the last sequence number before the sequence number wraps to 1, where *seqwrap* is a number from 100 to 999 999 999.

This number is nonnegotiable and must match in both the local and remote channel definitions.

SOCKBUF-*bufsize*
changes the socket buffer size, where *bufsize* is a value from 32 KB to 1024 KB.

XMITQ-*xmitq*
changes the transmission queue name (the queue from which messages are retrieved by the channel), where *xmitq* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

This parameter is valid only for sender channels. Any changes to the transmission queue name associated with a channel can be done only while the channel is stopped and not in an in-doubt condition.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example changes a channel definition.

```
User:   ZMQSC ALT CHL-'TPF.to.MVS' MAXMSGL-16000 XMITQ-'mymvs'

System: MQSC0006I 07.38.46 ALTER CHANNEL DEFINITION SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

See *TPF Application Programming* for more information about the TPF MQSeries ROUTC bridge.

ZMQSC ALT MQP–Alter a TPF MQSeries Profile

Use this command to change and set default values for a local MQSeries queue manager on your TPF system.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC ALT── ──MQP-qmanager──┬────────────────────┬──────────────────────────►
                                └─ ──DEADQ-dlqname─┘

►─┬─────────────────────┬─┬──────────────────────┬─┬───────────────────────┬─►
  └─ ──MAXMSGL-maxmsgl─┘  └─ ──QDEPTHHI-qdepthhi─┘  └─ ──BATCHSZ-batchsz─┘

►─┬───────────────────────┬─┬────────────┬─┬──────────────────────────┬────►
  └─ ──NEWNAME-qmanager─┘  └─ ──QDT-qdt─┘  └─ ──MAXDEPTH-maxdepth─┘

►─┬──────────────────────┬───────────────────────────────────────────────►◄
  └─ ──SWEEP-──┬─ON──┬─┘
               └─OFF─┘
```

MQP-*qmanager*
: specifies the name of the TPF MQSeries queue manager to be changed, where *qmanager* is the 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *qmanager* is enclosed in single quotation marks, the characters can be mixed case.

  You can have one queue manager in the TPF system complex for each subsystem.

DEADQ-*dlqname*
: defines the default dead-letter queue name, where *dlqname* is the name of the file that is used as the dead-letter queue.

MAXMSGL-*maxmsgl*
: changes the default maximum message length that can be transmitted on the channel or the maximum message length for a message on a queue, where *maxmsgl* is a number in the range 0–4 194 304.

QDEPTHHI-*qdepthhi*
: changes the default percentage of messages that are allowed on subsequently

defined message queues before a warning message is sent to the console, where *qdepthhi* is a number from 0 to 100.

If you specify 0, no warning messages are sent to the console.

BATCHSZ-*batchsz*
changes the default maximum number of messages that can be sent over any subsequently defined channels before TPF MQSeries guarantees delivery of the messages in the batch. The TPF batch size is from 1 to 9999 messages. The maximum batch size actually used is the lowest of the following:

- The batch size of the sending channel
- The batch size of the receiving channel.

NEWNAME-*qmanager*
specifies the new name of the TPF MQSeries queue manager in the TPF system, where *qmanager* is the 1- to 48-character alphanumeric name. If *qmanager* is enclosed in single quotation marks, the characters can be mixed case.

You can have one queue manager in the TPF system complex for each subsystem.

The new name may not be the same as any RQMNAME for any defined or remote queues.

QDT-*qdt*
specifies the time interval, in seconds, for checking the maximum queue depth that is set for each transmission queue, where *qdt* is a number from 0 to 999 999 999.

If you specify 0, the queue depth is not checked.

MAXDEPTH-*maxdepth*
changes the default maximum number of messages allowed on all subsequently defined memory queues, where *maxdepth* is a number from 0 to 999 999 999. If you specify zero, there is no maximum for any subsequently defined queues.

Other factors can cause the queue to be handled as full, for example, if there is no more DASD or memory space available.

SWEEP
specifies one of the following:

ON
specifies that memory queues are moved (swept) to a TPF collection support (TPFCS) persistent collection when the number of messages on the queue are increasing.

OFF
specifies that memory queues are not swept.

Attention: Setting the SWEEP parameter to OFF can deplete system work blocks (SWBs).

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example changes a TPF MQSeries profile.

```
User:   ZMQSC ALT MQP-TPFMQA MAXMSGL-4000

System: MQSC0002I 08.41.25 ALTER PROFILE SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC ALT PROCESS–Alter a TPF MQSeries Process

Use this command to change an MQSeries process on your TPF system.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC ALT── ──PROCess-processname──┬───────────────────────┬──────────────►
                                        └─ ──APPLICID-appname──┘

►──┬─────────────────────┬──┬────────────────────────┬──┬────────────────────────┬──►◄
   └─ ──DESCR-comment────┘  └─ ──ENVRDATA-envrdata───┘  └─ ──USERDATA-userdata───┘
```

PROCESS-*processname*
: specifies the name of a process object that is to be changed on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

APPLICID-*appname*
: specifies the name of the TPF application program to be started, where *appname* is a 4-character TPF program name.

DESCR-*comment*
: specifies a plaintext comment that provides descriptive information about the object when you enter the ZMQCS DISPLAY command with the PROCESS parameter specified, where *comment* is a 1- to 64-character plaintext comment.

ENVRDATA-*envrdata*
: specifies a character string that contains environment information pertaining to the application to be started, where *envrdata* is a 1- to 128-character string.

USERDATA-*userdata*
: specifies a character string that contains user information pertaining to the application specified by the APPLICID parameter that is to be started, where *userdata* is a 1- to 128-character string.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example changes the CREDITCARD process.

```
User:   ZMQSC ALT PROCESS-CreditCard APPLICID-ABCD

System: MQSC0701I 09.17.22 PROCESS CREDITCARD DEFINITION ALTERED SUCCESSFULLY
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC ALT QA–Alter a TPF MQSeries Alias Queue Definition

Use this command to change a TPF MQSeries alias queue definition.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC ALT── ──QA-qaname──┬──────────────┬──┬─────────────────────┬──────────►
                             └─ ──TARGQ-tname─┘  └─ ──PUT-─┬─ENABLED──┬─┘
                                                           └─DISABLED─┘

►──┬─────────────────────┬──┬───────────────────────┬──────────────────────────►◄
   └─ ──GET-─┬─ENABLED──┬─┘  └─ ──DEFPSIST-─┬─NO──┬─┘
             └─DISABLED─┘                   └─YES─┘
```

QA-*qaname*
: specifies the name of the alias queue to be changed, where *qaname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

TARGQ-*tname*
: specifies the name of the queue defined with the ZMQSC DEF QL or ZMQSC DEF QR command to which this alias will refer, where *tname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

PUT
: specifies one of the following:

  ENABLED
  : specifies that applications can add messages to the queue.

  DISABLED
  : specifies that applications cannot add messages to the queue.

GET
: specifies one of the following:

  ENABLED
  : specifies that applications can retrieve messages from the queue.

  DISABLED
  : specifies that applications cannot retrieve messages from the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example changes an alias queue.

```
User:   ZMQSC ALT QA-'aq2' GET-DISABLED

System: MQSC0148I 08.41.05 ALIAS QUEUE aq2 ALTERED
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC ALT QL–Alter a TPF MQSeries Local Queue Definition

Use this command to change a TPF MQSeries local queue definition.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC ALT── ──QL-qlname──┬────────────────────────┬──────────────────►
                             └─ ──MAXMSGL-maxmsgl─┘

►──┬──────────────────────────┬─┬──────────────────────────┬──────────────►
   └─ ──PUT-──┬─ENABLED──┬──┘ └─ ──GET-──┬─ENABLED──┬──┘
              └─DISABLED─┘               └─DISABLED─┘

►──┬───────────────────────┬─┬──────────────────────────┬─────────────────►
   └─ ──DEFPSIST-──┬─NO──┬─┘ └─ ──QDEPTHHI-qdepthhi─┘
                   └─YES─┘

►──┬───────────────────────────┬─┬──────────────────────┬─────────────────►
   └─ ──MAXDEPTH-maxdepth─┘ └─ ──SWEEP-──┬─ON──┬─┘
                                         └─OFF─┘

►──┬────────────────────────────┬─┬──────────────────────────┬────────────►
   └─ ──TRIGTYPE-──┬─FIRST─┬─┘ └─ ──TRIGDATA-trigdata─┘
                   ├─EVERY─┤
                   └─NONE──┘

►──┬───────────────────────┬─┬────────────────────────────────┬─────────►◄
   └─ ──┬─NOTRIGGER─┬─┘ └─ ──PROCESS-processname─┘
        └─TRIGGER───┘
```

QL-*qlname*
: specifies the name of the local queue to be changed, where *qlname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

MAXMSGL-*maxmsgl*
changes the maximum message length on this local queue, where *maxmsgl* is a number from 0 to the maximum message length specified in the TPF MQSeries profile.

Applications can use this attribute to determine the size of the buffer they need to retrieve messages from the queue.

PUT
specifies one of the following:

ENABLED
specifies that applications can add messages to the queue.

DISABLED
specifies that applications cannot add messages to the queue.

GET
specifies one of the following:

ENABLED
specifies that applications can retrieve messages from the queue.

DISABLED
specifies that applications cannot retrieve messages from the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

QDEPTHHI-*qdepthhi*
changes the default percentage of messages that are allowed before a warning message is sent to the console, where *qdepthhi* is a number from 0 to 100.

If you specify 0, no warnings are sent to the console.

This parameter is not valid for queues that are processor shared; that is, queues defined with the COMMON parameter set to YES.

MAXDEPTH-*maxdepth*
changes the maximum number of messages allowed on a memory queue, where *maxdepth* is a number from 0 to 999 999 999. If you specify zero, there is no maximum for this queue.

Other factors can cause the queue to be handled as full; for example, if there is no more DASD or memory space available.

This parameter is not valid for queues that are processor shared; that is, queues defined with the COMMON parameter set to YES.

SWEEP
specifies one of the following:

ON
specifies that this memory queue is moved (swept) to a TPF collection support (TPFCS) persistent collection when the number of messages on the queue are increasing.

OFF
specifies that this memory queue is not swept.

Attention: Setting the SWEEP parameter to OFF can decrease the number of system work blocks (SWBs).

TRIGTYPE
specifies one of the following:

FIRST
specifies that the program in the process object is triggered to run the first time a message arrives on the queue. If no process is associated with the queue, the TPF queue trigger user exit (CUIR) is called.

EVERY
specifies that the program in the process object is triggered to run every time a message arrives on the queue. If no process is associated with the queue, no triggering occurs.

NONE
specifies that the program in the process object is not triggered to run.

TRIGDATA-*trigdata*
specifies a character string that contains trigger information pertaining to the application to be started, where *trigdata* is a 1- to 64-character string. This is free-format data that the queue manager inserts into the trigger message when a message arriving on this queue causes a trigger message to be written.

NOTRIGGER
specifies that triggering is not active.

TRIGGER
specifies that triggering is active and will occur as specified by the TRIGTYPE parameter.

PROCESS-*processname*
specifies the name of a process object that is defined on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?

Examples

The following example changes a local queue.

```
User:   ZMQSC ALT QL-'local.queue.' GET-DISABLED

System: MQSC0146I 08.41.05 LOCAL QUEUE local.queue ALTERED
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC ALT QR–Alter a TPF MQSeries Remote Queue Definition

Use this command to change a TPF MQSeries remote queue definition.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC ALT── ──QR-qrname──┬──────────────────┬─┬──────────────┬──────────►
                             └─ ──RQMNAME-rqmname─┘ └─ ──RNAME-rname─┘

►──┬────────────────┬─┬────────────────────┬─┬─────────────────────┬──────►◄
   └─ ──XMITQ-xmitq─┘ └─ ──PUT-──┬─ENABLED──┬┘ └─ ──DEFPSIST-──┬─NO──┬┘
                                 └─DISABLED─┘                  └─YES─┘
```

QR-*qrname*
: specifies the name of the remote queue definition to be changed, where *qrname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

RQMNAME-*rqmname*
: changes the name of the remote queue manager on which the remote queue is defined, where *rqmname* is a 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

RQMNAME cannot be the same as the local queue manager name.

RNAME-*rname*
: changes the name of the remote queue, where *rname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

Notes:

1. Specify blanks enclosed in single quotation marks (' ') for this parameter to change a previously defined local definition for a remote queue to a queue manager alias.
2. Specify a queue name for this parameter to change a previously defined queue manager alias to a local definition for a remote queue or to change the name of the remote queue in the local definition of the remote queue.

XMITQ-*xmitq*
changes the name of the transmission queue used for forwarding messages to the remote queue, where *xmitq* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. If you specify blanks (' ') for the name, the remote queue manager name specified with the RQMNAME parameter is used.

PUT
specifies one of the following:

ENABLED
specifies that applications can add messages to the queue.

DISABLED
specifies that applications cannot add messages to the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZMQSC HELP
ZMQSC ?

Examples

The following example changes a remote queue.

```
User:   ZMQSC ALT QR-OS2Q XMITQ-H2

System: MQSC0149I 08.41.05 REMOTE QUEUE OS2Q ALTERED
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC CLEAR QL–Remove Messages from a TPF MQSeries Local Queue

Use this command to remove all messages from a local normal queue.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before you enter this command.
- You cannot clear a transmission queue that is in use by an active channel.
- For processor unique queues, the queue is cleared only on the processor on which you enter the command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC CLEAR── ──QL-qlname──┬────────┬──────────────────────────►◄
                               └─FORCE──┘
```

QL-*qlname*
: specifies the name of the local queue to be cleared, where *qlname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

FORCE
: clears a queue even if there were errors when removing messages from the queue.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example clears the LOCALQ queue:

```
User:   ZMQSC CLEAR QL-LOCALQ

System: MQSC0219I 08.45.03 ALL MESSAGES REMOVED FROM LOCAL QUEUE LOCALQ
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DBREBUILD–Rebuild TPF MQSeries Definitions

Use this command to rebuild TPF MQSeries definitions in the current file address reference format (FARF) on the TPF system without losing those definitions and without losing any messages that are currently on queue.

Note: To migrate a TPF MQSeries data store (for example, MQSCBSS), enter the ZOODB MIGRATE command before or after you enter the ZMQSC DBREBUILD command.

Requirements and Restrictions

- Ensure that only one processor is active in a loosely coupled environment.
- Ensure that the TPF MQSeries queue manager is stopped on the subsystem from which you enter this command. See “ZMQSC STOP–Stop a TPF MQSeries Channel or Queue Manager” on page 907 for more information .
- Enter this command on each subsystem where a TPF MQSeries queue manager has been defined.
- Do not start a TPF MQSeries queue manager on a subsystem from which this command is running. See “ZMQSC START–Start a TPF MQSeries Channel or Queue Manager” on page 905 for more information.
- Do not activate a processor in a loosely coupled environment while this command is running.
- You can enter this command only in NORM state.

Format

```
►►──ZMQSC DBREBUILD──┬─────────┬──────────────────────────────►◄
                     └──FORCE──┘
```

FORCE
rebuilds TPF MQSeries definitions, but removes any nonessential TPF MQSeries definitions (such as, TPF MQSeries trace settings) that TPF could not build because errors occurred.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZMQSC HELP
ZMQSC ?

Examples

The following command can be used to rebuild your MQSeries definitions:

```
User:   ZMQSC DBREBUILD

System: MQSC0900I 08:14:31 DBREBUILD COMPLETED SUCCESSFULLY
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DEF CHL–Define a TPF MQSeries Channel

Use this command to define a TPF MQSeries channel. A channel is needed for the TPF MQSeries queue manager to deliver a message to or receive a message from a remote queue.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                               ┌─ ──CPU-localid────┐
►►──ZMQSC DEF── ──CHL-name─────┤                   ├──────────────────────────────────────►
                               └─ ──CPU-──┬cpuid─┬─┘
                                          └ALL───┘

►─ ──CHLTYPE-──┬SDR──┤ Sender Channel Parameters ├────────────────────────┬──────────►◄
               ├RCVR──┤ Receiver Channel Parameters ├─────────────────────┤
               └SVRCONN──┤ Server Connection Channel Parameters ├─────────┘
```

```
Sender Channel Parameters:

                                                        ┌─ ──DESCR-'    '──┐
├──┬─────────────────────────┬─ ──CONNAME-conname───────┤                  ├──────────►
   └─ ──BATCHSZ-batchsz──────┘                          └─ ──DESCR-descr───┘

                                                             ┌─ ──MSGDATA-'    '────┐
►──┬───────────────────────────┬──┬─────────────────────┬────┤                      ├──►
   └─ ──LOCALIP-ipaddress──────┘  └─ ──MAXMSGL-maxmsgl──┘    └─ ──MSGDATA-msgdata───┘

   ┌─ ──MSGEXIT-NO───┐
►──┤                 ├──┬────────────────────────────┬──┬──────────────────────────┬────►
   └─ ──MSGEXIT-YES──┘  └─ ──NPMSPEED-──┬FAST───┬───┘  └─ ──SOCKBUF-bufsize──────┘
                                        └NORMAL─┘

                                  ┌─ ──SCYEXIT-NO───┐
►──┬───────────────────────────┬──┤                 ├──┬────────────────────────┬────────►
   └─ ──SCYDATA-scydata────────┘  └─ ──SCYEXIT-YES──┘  └─ ──SEQWRAP-seqwrap─────┘

►─ ──XMITQ-xmitq─────────────────────────────────────────────────────────────────────────┤
```

Receiver Channel Parameters:

```
                                  ┌─ ──DESCR-'    '─┐
├──┬──────────────────────┬───────┼─────────────────┼──┬────────────────────────┬──────────►
   └─ ──BATCHSZ-batchsz───┘       └─ ──DESCR-descr──┘  └─ ──MAXMSGL-maxmsgl─────┘

   ┌─ ──MRDATA-'    '─┐  ┌─ ──MREXIT-NO──┐  ┌─ ──MSGDATA-'    '─┐
►──┼──────────────────┼──┼───────────────┼──┼───────────────────┼──────────────────────►
   └─ ──MRDATA-mrdata─┘  └─ ──MREXIT-YES─┘  └─ ──MSGDATA-msgdata─┘

   ┌─ ──MSGEXIT-NO──┐
►──┼────────────────┼──┬─────────────────────────────┬──┬──────────────────────┬──────►
   └─ ──MSGEXIT-YES─┘  └─ ──NPMSPEED-──┬─FAST───┬────┘  └─ ──SOCKBUF-bufsize───┘
                                       └─NORMAL─┘

                                  ┌─ ──SCYEXIT-NO──┐
►──┬──────────────────────────┬───┼────────────────┼──┬──────────────────────┬──────────┤
   └─ ──SCYDATA-scydata───────┘   └─ ──SCYEXIT-YES─┘  └─ ──SEQWRAP-seqwrap───┘
```

Server Connection Channel Parameters:

```
   ┌─ ──DESCR-'    '─┐
├──┼─────────────────┼──┬────────────────────────┬──┬───────────────────────┬─────────►
   └─ ──DESCR-descr──┘  └─ ──MAXMSGL-maxmsgl─────┘  └─ ──SCYDATA-scydata────┘

   ┌─ ──SCYEXIT-NO──┐
►──┼────────────────┼──┬──────────────────────┬─────────────────────────────────────────┤
   └─ ──SCYEXIT-YES─┘  └─ ──SOCKBUF-bufsize───┘
```

CHL-*name*
: specifies the name of the channel definition, where *name* is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

  Specify a name that is unique in the TPF MQSeries queue manager at the local host. Do ***not*** specify a channel name of ALL.

CHLTYPE-*chltype*
: defines one of the following channel types:

  SDR
  : specifies the sender channel.

  RCVR
  : specifies the receiver channel.

  SVRCONN
  : specifies the server connection channel.

BATCHSZ-*batchsz*
specifies the maximum number of messages that can be sent over a channel before TPF MQSeries guarantees delivery of the messages in the batch. The TPF maximum batch size range is from 1 to 9999 messages.

If you do not specify this parameter, the batch size defined with the ZMQSC DEF MQP command is used. The maximum batch size is 9999 messages. The maximum batch size actually used is the lowest of the following:

- The batch size of the sending channel
- The batch size of the receiving channel.

CONNAME-*conname*
defines the connection name, where *conname* is from 1 to 256 characters. This is either the host name (alphanumeric characters and periods) or the host Internet Protocol (IP) address (numeric characters and dotted decimal notation).

If the channel connects to a port number other than the well-known port 1414, this port number can be specified in parentheses at the end of *conname*, and *conname* must be enclosed in single quotation marks.

CPU
defines the message channel definition for the specified central processing unit (CPU), where:

cpuid
is a 1-character CPU identifier (ID).

ALL
defines all the processors in the system.

localid
is the local CPU ID from which the command is currently running. This is the default.

| **DESCR-***descr*
| specifies a description of the channel, where *descr* is from 1 to 64 alphanumeric characters. If *descr* is enclosed in single quotation marks, the characters can be mixed case.

| **LOCALIP-***ipaddress*
| specifies which local IP address is used when establishing a connection, where *ipaddress* is a numeric IP address. If this parameter is not specified, the default local IP address is used.

MAXMSGL-*maxmsgl*
defines the maximum message length that can be transmitted on the channel, where *maxmsgl* is a number from 0 to the maximum message length specified in the TPF MQSeries profile. If you do not specify this parameter, the maximum message length defined with the ZMQSC DEF MQP command is used.

| **MRDATA-***mrdata*
| specifies the data that will be passed to the channel message retry user exit in segment CUIT, where *mrdata* is from 1 to 32 alphanumeric characters. If *mrdata* is enclosed in single quotation marks, the characters can be mixed case.

| **MREXIT**
| specifies one of the following:

| **YES**
| specifies that the channel message retry user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel message retry user exit.

NO
specifies that the channel message retry user exit in segment CUIT will not be called.

MSGDATA-*msgdata*
specifies the data that will be passed to the channel message user exit in segment CUIT, where *msgdata* is from 1 to 32 alphanumeric characters. If *msgdata* is enclosed in single quotation marks, the characters can be mixed case.

MSGEXIT
specifies one of the following:

YES
specifies that the channel message user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel message user exit.

NO
specifies that the channel message user exit in segment CUIT will not be called.

NPMSPEED
defines the channel speed, where:

FAST
specifies the following:

- For sender channels, both persistent and nonpersistent messages are sent. Delivery is guaranteed for persistent messages only.
- For receiver channels, both nonpersistent messages and persistent messages are received. Receipt by TPF MQSeries is guaranteed only for persistent messages. Nonpersistent messages received on fast receiver channels are given to the TPF MQSeries ROUTC bridge for processing.

NORMAL
specifies the following:

- For sender channels, both persistent and nonpersistent messages are processed by normal sender channels. Delivery is guaranteed for both persistent and nonpersistent messages.
- For receiver channels, receipt by TPF MQSeries for all messages (persistent and nonpersistent) is guaranteed.

SEQWRAP-*seqwrap*
specifies the last sequence number before the sequence number wraps to 1, where *seqwrap* is a number from 100 to 999 999 999.

This number is nonnegotiable and must match in both the local and remote channel definitions.

SCYEXIT
specifies one of the following:

YES
specifies that the channel security user exit in segment CUIT will be called. See *TPF System Installation Support Reference* for more information about the channel security user exit.

NO
specifies that the channel security user exit in segment CUIT will not be called.

SCYDATA-*scydata*
specifies the data that will be passed to the channel security user exit in segment CUIT , where *scydata* is from 1 to 32 alphanumeric characters. If *scydata* is enclosed in single quotation marks, the characters can be mixed case.

SOCKBUF-*bufsize*
specifies the socket buffer size, where *bufsize* is a value from 32 KB to 1024 KB. If this parameter is not specified, the default socket buffer size is used.

XMITQ-*xmitq*
defines the transmission queue name (the name of the queue from which messages are retrieved by the channel), where *xmitq* is a 1- to 48-characters queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example creates a channel definition.

```
User:   ZMQSC DEF CHL-TPF.to.OS2 CHLTYPE-SDR CONNAME-TPF.POK.IBM.COM XMITQ-MYOS2

System: MQSC0004I 08.45.03 DEFINE CHANNEL SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

See *TPF Application Programming* for more information about the TPF MQSeries ROUTC bridge.

ZMQSC DEF MQP–Define a TPF MQSeries Profile

Use this command to define and set default values for a local MQSeries queue manager on your TPF system.

Requirements and Restrictions

- You can enter this command only in NORM state.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                                   ┌─ ──DEADQ-DEAD.LETTER.QUEUE─┐
►►──ZMQSC DEF── ──MQP-qmanager─────┼────────────────────────────┼──────────────────►
                                   └─ ──DEADQ-dlqname───────────┘

  ┌─ ──MAXMSGL-10240──┐  ┌─ ──QDEPTHHI-0──────────┐  ┌─ ──BATCHSZ-10──────┐
►─┼───────────────────┼──┼────────────────────────┼──┼────────────────────┼──────►
  └─ ──MAXMSGL-maxmsgl┘  └─ ──QDEPTHHI-qdepthhi───┘  └─ ──BATCHSZ-batchsz──┘

  ┌─ ──QDT-0───┐  ┌─ ──MAXDEPTH-5000──────┐
►─┼────────────┼──┼───────────────────────┼──┬──────────────────────┬───────────►◄
  └─ ──QDT-qdt─┘  └─ ──MAXDEPTH-maxdepth──┘  └─ ──SWEEP-─┬─ON──┬─────┘
                                                         └─OFF─┘
```

MQP-*qmanager*
: specifies the name of the TPF MQSeries queue manager, where *qmanager* is the 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

  There is only one queue manager for every TPF subsystem.

DEADQ-*dlqname*
: defines the default dead-letter queue name, where *dlqname* is the name of the file that is used as the dead-letter queue.

MAXMSGL-*maxmsgl*
: defines the default maximum message length that can be transmitted on subsequently defined channels or the maximum message length for a message on subsequently defined queues, where *maxmsgl* is a number in the range 0–4 194 304.

QDEPTHHI-*qdepthhi*
: specifies the default percentage of messages that are allowed on a queue before a warning message is sent to the console, where *qdepthhi* is a number from 0 to 100.

  If you specify 0, no warning messages are sent to the console.

BATCHSZ-*batchsz*
: specifies the default maximum number of messages that can be sent over any

subsequently defined channels before TPF MQSeries guarantees delivery of the messages in the batch, where *batchsz* is the TPF batch size from 1 to 9999 messages. The maximum batch size actually used is the lowest of the following:

- The batch size of the sending channel
- The batch size of the receiving channel.

QDT-*qdt*
specifies the time interval, in seconds, for checking the maximum queue depth that is set for each transmission queue, where *qdt* is a number from 0 to 999 999 999.

If you specify 0, the maximum queue depth is not checked.

MAXDEPTH-*maxdepth*
defines the default maximum number of messages allowed on subsequently defined memory queues, where *maxdepth* is a number from 0 to 999 999 999. If you specify zero, there is no default maximum for subsequently defined queues.

Other factors can cause the queue to be handled as full, for example, if there is no more DASD or memory space available.

SWEEP
specifies one of the following:

ON
specifies that memory queues are moved (swept) to a TPF collection support (TPFCS) persistent collection when the number of messages on the queue are increasing.

OFF
specifies that memory queues are not swept.

Attention: Setting the SWEEP parameter to OFF can deplete system work blocks (SWBs).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- The TPF MQSeries profile is defined to all processors in the loosely coupled complex.

Examples

The following example defines the TPF MQSeries profile.

```
User:   ZMQSC DEF MQP-TFFQMGR

System: MQSC0001I 08.41.25 DEFINE PROFILE SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DEF PROCESS–Define a TPF MQSeries Process

Use this command to define an MQSeries process on your TPF system.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                                         ┌─ ──APPLICID-'    '──┐
►►──ZMQSC DEF── ──PROCess-processname────┤                     ├──────────────►
                                         └─ ──APPLICID-appname─┘

                                ┌─ ──DESCR-'    '──┐ ┌─ ──ENVRDATA-'    '───┐
►─┬─────────────────────┬───────┤                  ├─┤                       ├──►
  └─ ──APPLTYPE-TPF─────┘       └─ ──DESCR-comment─┘ └─ ──ENVRDATA-envrdata──┘

  ┌─ ──USERDATA-'    '───┐
►─┤                      ├─────────────────────────────────────────────────────►◄
  └─ ──USERDATA-userdata─┘
```

PROCESS-*processname*
: specifies the name of a process object that is defined on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

Note: Do not specify ALL as a process name.

APPLICID-*appname*
: specifies the name of the TPF MQSeries program to be started, where *appname* is a 4-character TPF program name.

APPLTYPE-TPF
: specifies the type of application to be started. TPF is the only supported application type.

DESCR-*comment*
: specifies a plaintext comment that provides descriptive information about the object when you enter the ZMQCS DISPLAY command with the PROCESS parameter specified, where *comment* is a 1- to 64-character plaintext comment.

ENVRDATA-*envrdata*
: specifies a character string that contains environment information pertaining to the application to be started, where *envrdata* is a 1- to 128-character string.

USERDATA-*userdata*
specifies a character string that contains user information pertaining to the application specified by the APPLICID parameter that is to be started, where *userdata* is a 1- to 128-character string.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example defines the CREDITCARD process to the TPF system.

```
User:   ZMQSC DEF PROCESS-CreditCard APPLICID-CDEF

System: MQSC0700I 09.17.22 PROCESS CREDITCARD DEFINITION COMPLETE SUCCESSFULLY
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DEF QA–Define a TPF MQSeries Alias Queue

Use this command to define a TPF MQSeries alias queue. An alias queue is a virtual queue that is resolved to another queue name.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                                                 ┌─ ──PUT-ENABLED──┐
►►──ZMQSC DEF── ──QA-qaname── ──TARGQ-tname──┤                  ├──────────────►
                                                 └─ ──PUT-DISABLED─┘

  ┌─ ──GET-ENABLED──┐
►─┤                  ├──┬─────────────────────────┬──────────────────────────►◄
  └─ ──GET-DISABLED─┘  └─ ──DEFPSIST-─┬─NO──┬──┘
                                         └─YES─┘
```

QA-*qaname*
specifies the name of the alias queue to be defined, where *qaname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be unique in the entire TPF system complex.

Do ***not*** specify an alias queue name of ALL.

TARGQ-*tname*
specifies the name of the queue defined with the ZMQSC DEF QL or ZMQSC DEF QR command to which this alias will refer, where *tname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

PUT
specifies one of the following:

ENABLED
specifies that applications can add messages to the queue.

Note: Messages will only be added to the queue if the definition for the target queue also specifies PUT-ENABLED.

DISABLED
specifies that applications cannot add messages to the queue.

GET
specifies one of the following:

ENABLED
specifies that applications can retrieve messages from the queue.

Note: Messages will only be retrieved from the queue if the definition for the target queue also specifies GET-ENABLED.

DISABLED
specifies that applications cannot retrieve messages from the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZMQSC HELP
ZMQSC ?

Examples

The following example defines an alias queue named AQ1, which refers to local queue LQ1. Receiving messages is disabled for this queue. Adding messages to the queue is enabled if the ability to add messages is enabled for local queue LQ1.

```
User:   ZMQSC DEF QA-AQ1 TARGQ-LQ1 GET-DISABLED

System: MQSC0143I 08.42.06 ALIAS QUEUE AQ1 DEFINED
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

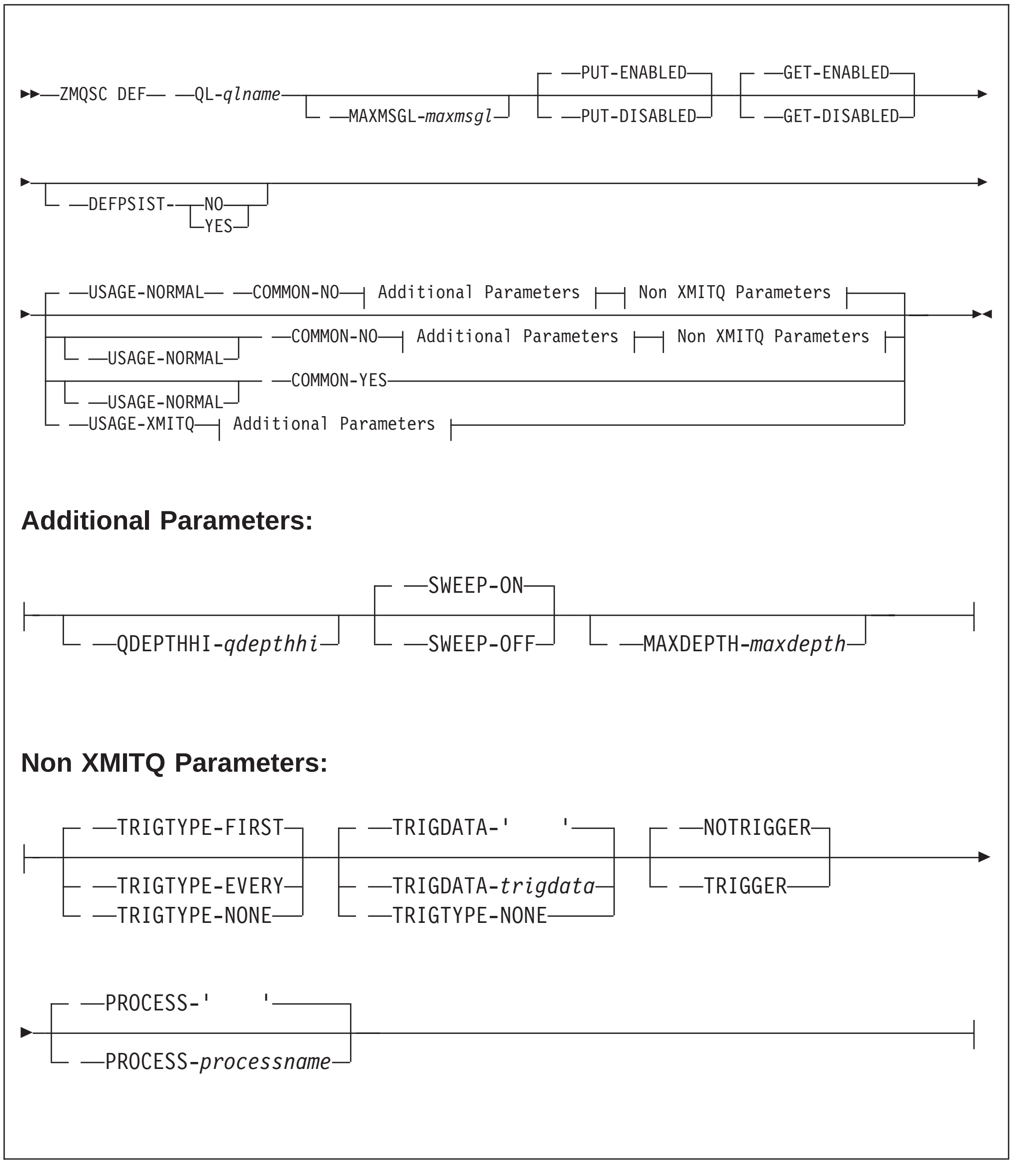
ZMQSC DEF QL–Define a TPF MQSeries Local Queue

Use this command to define a TPF MQSeries local queue. A local queue must be defined so that an application can get messages from the queue or put messages on the queue.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format



QL-*qlname*
specifies the local name of the queue, where *qlname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be unique in the entire TPF system complex.

Do ***not*** specify a local queue name of ALL.

MAXMSGL-*maxmsgl*
defines the maximum message length on this local queue, where *maxmsgl* is a number from 0 to the maximum message length specified in the TPF MQSeries profile. If you do not specify this parameter, the maximum message length defined with the ZMQSC DEF MQP command is used.

You can use this to determine the size of the buffer that an application needs to retrieve messages from the queue.

PUT
specifies one of the following:

ENABLED
specifies that messages can be added to the queue.

DISABLED
specifies that messages cannot be added to the queue.

GET
specifies one of the following:

ENABLED
specifies that messages can be retrieved from the queue.

DISABLED
specifies that applications cannot retrieve messages from the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

USAGE
defines the type of queue, where:

NORMAL
indicates that the queue is not a transmission queue.

XMITQ
indicates that the queue is a transmission queue.

COMMON
specifies whether the physical queue is shared, where:

NO
specifies that each central processing unit (CPU) has its own physical queue. A queue defined with this value is a processor unique queue.

YES
specifies that all CPUs share the same physical queue. A queue defined with this value is a processor shared queue.

QDEPTHHI-*qdepthhi*
specifies the default percentage of messages that are allowed for the local queue before a warning message is sent to the console, where *qdepthhi* is a number from 0 to 100.

If you do not specify this parameter, the value that was specified with the QDEPTHHI parameter of the ZMQSC DEF MQP command is used. If you specify 0, no warning messages are sent to the console.

This parameter is not valid for queues that are processor shared; that is, queues defined with the COMMON parameter set to YES.

SWEEP
specifies one of the following:

ON
specifies that this memory queue is moved (swept) to a TPF collection support (TPFCS) persistent collection when the number of messages on the queue are increasing.

OFF
specifies that this memory queue is not swept.

Attention: Setting the SWEEP parameter to OFF can deplete system work blocks (SWBs).

MAXDEPTH-*maxdepth*
changes the maximum number of messages allowed on this memory queue, where *maxdepth* is a number from 0 to 999 999 999. If you specify zero, there is no maximum for this queue.

Other factors can cause the queue to be handled as full; for example, if there is no more DASD or memory space available.

If you do not specify this parameter, the value that was specified for the MAXDEPTH parameter of the ZMQSC DEF MQP command is used.

This parameter is not valid for queues that are processor shared; that is, queues defined with the COMMON parameter set to YES.

TRIGTYPE
specifies one of the following:

FIRST
specifies that the program in the process object is triggered to run the first time a message arrives on the queue. If no process is associated with the queue, the TPF queue trigger user exit (CUIR) is called.

EVERY
specifies that the program in the process object is triggered to run every time a message arrives on the queue. If no process is associated with the queue, no triggering occurs.

NONE
specifies that the program in the process object is not triggered to run.

NOTRIGGER
specifies that triggering is not active.

TRIGGER
specifies that triggering is active and will occur as specified by the TRIGTYPE parameter.

TRIGDATA-*trigdata*
specifies a character string that contains trigger information pertaining to the application to be started, where *trigdata* is a 1- to 64-character string. This is free-format data that the queue manager inserts into the trigger message when a message arriving on this queue causes a trigger message to be written.

PROCESS-*processname*
specifies the name of a process object that is defined on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- The queue is defined to all processors in a loosely coupled complex.
- Processor unique queues reside in memory and are made persistent by filing a copy of the queue to fixed file records on a time-initiated basis, and all changes to the queue during a checkpoint interval are sent to the recovery log. Processor shared queues reside in TPF collection support (TPFCS) persistent collections.

Examples

The following example defines a new local queue.

```
User:   ZMQSC DEF QL-'target.queue' GET-DISABLED USAGE-NORMAL

System: MQSC0069I 08.41.05 LOCAL QUEUE target.queue DEFINED
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DEF QR–Define a TPF MQSeries Remote Queue

Use this command to define a local definition of a remote queue or a queue manager alias. A remote queue is one that is owned by another queue manager that applications connected to the TPF MQSeries queue manager need to access.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
>>──ZMQSC DEF── ──QR-qrname──┬─────────────────────┬─┬─────────────────┬──────────►
                             └──RQMNAME-rqmname────┘ └──RNAME-rname────┘

                         ┌──PUT-ENABLED──┐
►─┬──────────────────┬───┼───────────────┼─┬─────────────────────────┬──────────►◄
  └──XMITQ-xmitq─────┘   └──PUT-DISABLED─┘ └──DEFPSIST-─┬─NO──┬─────┘
                                                        └─YES─┘
```

QR-*qrname*
: specifies the name of the local definition of a remote queue or the queue manager alias, where *qrname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9).

  Do ***not*** specify a remote queue name of ALL.

RQMNAME-*rqmname*
: specifies the name of the remote queue manager on which the remote queue is defined, where *rqmname* is a 1- to 48-character queue manager name. A queue manager name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

  RQMNAME cannot be the same as the local queue manager name.

RNAME-*rname*
: specifies the name of the remote queue, where *rname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

  If you specify this parameter, a local definition for the remote queue is defined. If you do not specify this parameter, a queue manager alias is defined.

XMITQ-*xmitq*
: specifies the name of the transmission queue to be used for forwarding

messages to the remote queue, where *xmitq* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. If you do not specify this parameter, the value specified for the RQMNAME parameter is used instead.

PUT
specifies one of the following:

ENABLED
specifies that messages can be added to the queue.

DISABLED
specifies that messages cannot be added to the queue.

DEFPSIST
specifies one of the following:

NO
specifies that messages on the queue are lost when the queue manager is restarted.

YES
specifies that messages on the queue are retained when the queue manager is restarted.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- The queue is defined to all processors in a loosely coupled complex.

Examples

The following example defines a local definition for a remote queue called RQ1 and specifies that a transmission queue called H2 will be used.

```
User:   ZMQSC DEF QR-RQ1 RNAME-OS2Q RQMNAME-OS2MGR XMITQ-H2

System: MQSC0145I 08.41.05 REMOTE QUEUE RQ1 DEFINED
```

The following example defines a queue manager alias called QMGR1 and specifies that a transmission queue called H3 will be used.

```
User:   ZMQSC DEF QR-QMGR1 RQMNAME-OS2MGR XMITQ-H3

System: MQSC0145I 08.41.05 REMOTE QUEUE QMGR1 DEFINED
```

The following example defines a local definition for a remote queue called RQ2 and specifies that a transmission queue called OS2MGR will be used.

```
User:   ZMQSC DEF QR-RQ2 RQMNAME-OS2MGR RNAME-OS2Q

System: MQSC0145I 08.41.05 REMOTE QUEUE RQ2 DEFINED
```

Related Information

See *MQSeries Command Reference* for more information about defining local definitions of remote queues and queue manager aliases.

ZMQSC DEL–Delete a TPF MQSeries Resource

Use this command to delete a TPF MQSeries channel, alias queue, local queue, or remote queue.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC DEL── ──┬─PROCESS-processname───────────────┬──────────────────►◄
                  │            ┌─ ──CPU-localid───┐    │
                  ├─CHL-name───┤                  ├────┤
                  │            └─ ──CPU-─┬─cpuid─┬┘    │
                  │                      └─ALL───┘     │
                  ├─QL-qlname──┬──────────────────────┬┤
                  │            └─ PURGE-─┬─procid─┬───┘│
                  │                      └─ALL────┘    │
                  ├─QR-qrname──────────────────────────┤
                  └─QA-qaname──────────────────────────┘
```

PROCESS-*processname*
: specifies the name of a process object that is to be deleted on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

CHL-*name*
: specifies the name of the channel definition to be deleted, where *name* is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be that of an existing channel.

Note: Ensure the channel is stopped before you specify this parameter to delete the channel.

CPU
: deletes a TPF MQSeries channel from the specified central processing unit (CPU), where:

cpuid
: is a 1-character CPU identifier (ID).

ALL
deletes the channel from all the processors in the TPF system.

localid
is the local CPU ID from which the command is currently running. This is the default.

QL-*qlname*
specifies the name of a local queue to be deleted, where *qlname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be defined to the local TPF MQSeries queue manager.

PURGE
deletes a specified local queue even if there are still messages on that queue, where:

procid
is a 1-character processor identifier (ID) that is only valid for processor unique queues.

ALL
deletes the local queue on all processors and is required for queues defined with the COMMON parameter set to YES.

QR-*qrname*
specifies the remote queue to be deleted, where *qrname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be defined to the local TPF MQSeries queue manager.

QA-*qaname*
specifies the alias queue to be deleted, where *qaname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case. The name must be defined to the local TPF MQSeries queue manager.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- The resource is deleted from all processors in a loosely coupled complex, except when you specify a channel name. To delete a channel from all processors in a loosely coupled complex, you must specify the CPU parameter with a value of ALL.
- If the queue is not empty when you enter the ZMQSC DEL command with the QL parameter specified, the queue will be deleted when the queue becomes empty unless the PURGE parameter is specified.
- You cannot delete a transmission queue if it is associated with an active channel.

Examples

The following example deletes the CreditCard process.

```
User:    ZMQSC DEL PROCESS-'CreditCard'

System:  MQSC0723I 09.17.22 PROCESS-CreditCard DELETED SUCCESSFULLY
```

The following example deletes a TPF MQSeries resource definition (channel).

```
User:   ZMQSC DEL CHL-TPF.TO.OS2

System: MQSC0010I 08.41.05 DELETE CHANNEL DEFINITION SUCCESSFUL
```

The following example deletes a TPF MQSeries resource definition (local queue).

```
User:   ZMQSC DEL QL-LOCAL1

System: MQSC0607I 08.41.05 DELETION OF QUEUE LOCAL1 STARTED
```

The following example deletes a TPF MQSeries resource definition (remote queue).

```
User:   ZMQSC DEL QR-TPF.TO.OS2.QUEUE

System: MQSC0018I 08.41.05 DELETE REMOTE QUEUE SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC DISPLAY–Display TPF MQSeries Information

Use this command to display:

- Channel definitions
- Channel status
- TPF MQSeries profiles
- Queue definitions.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC DISplay── ──────────────────────────────────────────────────────►

►──┬─PROCESS──┬─processname──────┬───────────────────────────────────────┬──►◄
   │          ├─partprocessname──┤                                       │
   │          └─ALL──────────────┘                                       │
   ├─QMGR─────────────────────────────────────────────────────────────────┤
   │                                                    ┌──CPU-localid──┐ │
   ├─CHL──┬─name──────────────────────────────────────┬─┤               ├─┤
   │      ├─partname──┤ Channel Definition Parameters ├┤ └──CPU-cpuid───┘ │
   │      └─ALL──┤ Channel Definition Parameters ├─────┘                  │
   ├─CHS──┬─name──┤ Server Connection Channel Status Parameters ├─┬───────┤
   │      ├─partname──┤ Channel Status Parameters ├───────────────┤       │
   │      └─ALL──┤ Channel Status Parameters ├────────────────────┘       │
   ├─QL──┬─qlname─────────────────────────────────────────────────────────┤
   │     ├─partqlname──┤ Queue Local Parameters ├─┤                       │
   │     └─ALL──┤ Queue Local Parameters ├────────┘                       │
   ├─QR──┬─qrname─────────────────────────────────────────────────────────┤
   │     ├─partqrname──┤ Queue Remote Parameters ├─┤                      │
   │     └─ALL──┤ Queue Remote Parameters ├────────┘                      │
   ├─QA──┬─qaname─────────────────────────────────────────────────────────┤
   │     └─ALL──┤ Queue Alias Parameters ├─┘                              │
   └─Queue──┬─qname──────┬────────────────────────────────────────────────┘
            ├─partqname──┤
            └─ALL────────┘
```

Channel Definition Parameters:

```
├── ──┬──────────────┬───────────────────────────────────┤
      ├─AUTOSTART────┤
      ├─BATCHSZ──────┤
      ├─CHLTYPE──────┤
      ├─LOCALIP──────┤
      ├─MAXMSGL──────┤
      ├─MRDATA───────┤
      ├─MREXIT───────┤
      ├─MSGDATA──────┤
      ├─MSGEXIT──────┤
      ├─NPMSPEED─────┤
      ├─RCVR─────────┤
      ├─SCYDATA──────┤
      ├─SCYEXIT──────┤
      ├─SDR──────────┤
      ├─SOCKBUF──────┤
      ├─TRPTYPE──────┤
      └─XMITQ────────┘
```

Server Connection Channel Status Parameters:

```
├── ──┬──────────────────────────────┬────────────────────┤
      ├─SOCKNO-socknum───────────────┤
      ├─IPADDR-ALL───────────────────┤
      ├─IPADDR-host──────────────────┤
      └─IPADDR-host ─SOCKNO-socknum──┘
```

Channel Status Parameters:

```
├── ──┬─────────────┬──────────────────────────────────────────────┤
      ├─BATCHES─────┤
      ├─BATCHSZ─────┤
      ├─BUFSRCVD────┤
      ├─BUFSSENT────┤
      ├─BYTESRCVD───┤
      ├─BYTESSENT───┤
      ├─CHLTYPE─────┤
      ├─CONNAME─────┤
      ├─CRDATE──────┤
      ├─CURLUWID────┤
      ├─CURSEQNUM───┤
      ├─DATESTART───┤
      ├─INDOUBT─────┤
      ├─LOCALIP─────┤
      ├─LSTLUWID────┤
      ├─LSTMSGDAT───┤
      ├─LSTMSGTIM───┤
      ├─LSTSEQNUM───┤
      ├─MESSAGES────┤
      ├─NPMSPEED────┤
      ├─SEQWRAP─────┤
      ├─SOCKBUF─────┤
      ├─TIMESTART───┤
      └─XMITQ───────┘
```

Queue Local Parameters:

```
├── ─┬─CHNAME──────┬─────────────────────────────────────────────┤
     ├─COMMON──────┤
     ├─CRDATE──────┤
     ├─CRTIME──────┤
     ├─CURDEPTH────┤
     ├─CURSPID─────┤
     ├─DEFPSIST────┤
     ├─DELETED─────┤
     ├─DELPND──────┤
     ├─FIRSTSPID───┤
     ├─GET─────────┤
     ├─GETCOUNT────┤
     ├─INSTID──────┤
     ├─MAXDEPTH────┤
     ├─MAXMSGL─────┤
     ├─MEMORY──────┤
     ├─MODIFIED────┤
     ├─MSGLPNTRS───┤
     ├─MSGSWEPT────┤
     ├─NPERMSGS────┤
     ├─NUMMQCLOSE──┤
     ├─NUMMQOPEN───┤
     ├─PERMSGS─────┤
     ├─PROCESS─────┤
     ├─PUT─────────┤
     ├─QADDRESS────┤
     ├─QDEPTHHI────┤
     ├─QMGRNAME────┤
     ├─QSWEEP──────┤
     ├─SWEEP───────┤
     ├─SWEPTPID────┤
     ├─SWTARGET────┤
     ├─TRIGGER─────┤
     ├─TRIGDATA────┤
     ├─TRIGTYPE────┤
     └─USAGE───────┘
```

Queue Remote Parameters:

```
├──  ┬─────────┬─────────────────────────────────────────────┤
     ├─DEFPSIST─┤
     ├─DELETED──┤
     ├─GET──────┤
     ├─INSTID───┤
     ├─MODIFIED─┤
     ├─PUT──────┤
     ├─QMGRNAME─┤
     ├─RNAME────┤
     ├─RQMNAME──┤
     └─XMITQ────┘
```

Queue Alias Parameters:

```
├──  ┬─────────┬─────────────────────────────────────────────┤
     ├─DEFPSIST─┤
     ├─DELETED──┤
     ├─INSTID───┤
     ├─MODIFIED─┤
     ├─PUT──────┤
     ├─QMGRNAME─┤
     └─TARGQ────┘
```

PROCESS
specifies the name of the process object to be displayed, where:

processname
specifies the name of a process object that is defined on the local queue manager, where *processname* is the 1- to 48-character process name. A process name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If *processname* is enclosed in single quotation marks, the characters can be mixed case.

partprocessname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of process objects that are defined on the local queue manager. For example, to display information about all process objects beginning with WEB, enter:

ZMQSC DISPLAY PROCESS-WEB*

ALL
displays all the process names.

QMGR
displays the current TPF MQSeries profile.

CHL
specifies the name of the channel definition you want to display, where:

name
is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of channel names. For example, to display information about all channel names beginning with WEB, enter:

ZMQSC DISPLAY CHL-WEB*

ALL
displays all the channels for the specified central processing unit (CPU).

CPU
displays the message channel definitions or status for the specified CPU, where:

cpuid
is a 1-character CPU ID.

localid
is the local CPU ID from which the command is currently running. This is the default.

CHS
specifies the name of the channel definition for which status information is displayed, where:

name
is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of channel names. For example, to display information about all channel names beginning with WEB, enter:

ZMQSC DISPLAY CHS-WEB*

ALL
displays the current status for all channels.

QL
specifies the local queue definition to be displayed, where:

qlname
is a 1- to 48-character single local queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partqlname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of local queue names. For example, to display information about all local queue names beginning with WEB, enter:

ZMQSC DISPLAY QL-WEB*

ALL
displays all the local queues.

QR
specifies the remote queue definition to be displayed, where:

qrname
is a 1- to 48-character single remote queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partqrname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of remote queue names. For example, to display information about all remote queue names beginning with WEB, enter:

ZMQSC DISPLAY QR-WEB*

ALL
displays all the remote queues.

QA
specifies the queue alias definition to be displayed, where:

qaname
is a 1- to 48-character single alias queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partqaname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of alias queue names. For example, to display information about all alias queue names beginning with WEB, enter:

ZMQSC DISPLAY QA-WEB*

ALL
displays all the alias queues.

Queue
specifies the queue definition to be displayed, where:

qname
is a 1- to 48-character single queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

partqname
is the first few characters of a channel name followed by an asterisk (*) as a wildcard character to specify a group of queue names. For example, to display information about all queue names beginning with WEB, enter:

ZMQSC DISPLAY Q-WEB*

ALL
displays all the queues.

AUTOSTART
displays whether a channel starts automatically or not.

BATCHES
displays the total number of batches processed by a channel.

BATCHSZ
displays the negotiated batch size currently being used on a channel.

BUFSRCVD
displays the total number of buffers received from the remote channel.

BUFSSENT
displays the total number of buffers sent to the remote channel.

BYTESRCVD
displays the total number of bytes received from the remote channel.

BYTESSENT
displays the total number of bytes sent to the remote channel.

CHLTYPE
displays the type of channel that is being displayed.

CHNAME
displays a channel name of a local queue.

COMMON
displays whether a local queue is common or not.

CONNAME
displays the host Internet Protocol (IP) address or host name for the remote channel.

CRDATE
displays the date when a defined local queue was created.

CRTIME
displays the time when a defined local queue was created.

CURDEPTH
displays the total number of messages currently on a local queue.

CURLUWID
displays the logical unit of work identifier (LUWID) for the last message processed by a channel.

CURSEQNUM
displays the sequence number for the last message processed (sent or received) by a channel.

CURSPID
displays the current sweep persistent identifier (PID) for a local queue.

DATESTART
displays the date when a channel was started.

DEFPSIST
displays the default persistence for a queue.

DELETED
displays whether or not a queue has been deleted.

DELPND
displays whether there is a pending delete on a local queue.

FIRSTSPID
displays the first sweep PID for a local queue.

GET
displays whether or not applications can retrieve messages from a queue.

GETCOUNT
displays the MQGET calls that have been issued to a local queue.

INDOUBT
displays whether a channel is in an in-doubt condition or not.

INSTID
displays the instance ID of a queue.

IPADDR
displays the Internet Protocol (IP) address of one remote host or all remote hosts, where:

host
is a 1- to 256-character host name (alphanumeric characters and periods) or the host IP address (numeric characters and dotted decimal notation).

ALL
displays the IP address of all remote hosts.

LOCALIP
displays the local IP address that is used when establishing a connection.

LSTLUWID
displays the LUWID for the last message acknowledged.

LSTMSGDAT
displays the date when the last message was processed.

LSTMSGTIM
displays the time when the last message was processed.

LSTSEQNUM
displays the sequence number for the last message that has been acknowledged.

MAXDEPTH
displays the maximum number of messages allowed on a memory queue.

MAXMSGL
displays the maximum message length of a local queue.

MEMORY
displays whether or not a local queue is a memory queue.

MESSAGES
displays the total number of messages processed by a channel.

MODIFIED
displays whether or not a queue was modified

MRDATA
displays the data that will be passed to the channel message retry user exit.

MREXIT
displays whether the channel message retry user exit is called or not.

MSGDATA
displaysthe data that will be passed to the channel message user exit.

MSGEXIT
displays whether the channel message user exit is called or not.

MSGLPNTRS
displays information about the message list pointers for a local queue.

MSGSWEPT
displays the number of messages that were swept for a local queue.

NPERMSGS
displays the number of nonpersistent messages on a local queue.

NPMSPEED
displays the speed at which nonpersistent messages are transported on a channel.

NUMMQCLOSE
displays the number of closed queues on a local queue.

NUMMQOPEN
displays the number of open queues on a local queue.

PERMSGS
displays the number of persistent messages on a local queue.

PROCESS
displays the name of a process object that is defined on a local queue manager.

PUT
displays whether applications can add messages to a local queue or not.

QADDRESS
displays the queue address in system heap for a local queue.

QDEPTHHI
displays the percentage of messages that are allowed before a warning message is sent to the console.

QMGRNAME
displays the queue manager name for a queue.

QSWEEP
displays whether there is a queue sweep in progress for a local queue.

RCVR
displays all receiver channels.

RNAME
displays the name of a remote queue.

RQMNAME
displays the name of a remote queue manager.

SCYDATA
displays the data passed to the security exit.

SCYEXIT
displays whether the channel security exit is called or not.

SDR
displays all sender channels.

SEQWRAP
displays the sequence wrap value for a channel.

SOCKBUF
displays the socket buffer size.

SOCKNO-*socknum*
displays the channel instance of a socket number, where *socknum* is the 6- to 8-digit hexadecimal socket descriptor.

SWEEP
displays whether sweeping is on or off for a local queue.

SWEPTPID
displays the number of swept PIDs for a local queue.

SWTARGET
displays the swing target of a local queue.

TARGQ
displays the base queue name.

TIMESTART
displays the time when a channel was started.

TRIGGER
displays whether or not triggering is active for a local queue.

TRIGDATA
displays the data that the queue manager inserts into the trigger message when a message arrives on a local queue.

TRIGTYPE
displays the type of triggering used for a local queue.

TRPTYPE
displays the type of communication protocol for a channel.

USAGE
displays the usage of a local queue.

XMITQ
displays the transmission queue used by the sender channel.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example displays information about the CreditCard process .

```
User:    ZMQSC DISP PROCESS-'CreditCard'

System:  MQSC0702I 09.17.22 PROCESS CreditCard DISPLAY :
             APPLICID -  CDEF
             USERDATA - RUN CREDIT CHECK
             ENVIRDATA -
             DESCR -
             APPLTYPE - TPF
```

The following example displays the current status for channel TPF.TO.NT.1, where:

CHLTYPE
displays the type of the channel for which status is being displayed.

CONNAME
displays the host Internet Protocol (IP) address or host name for the remote channel.

NPMSPEED
displays the speed at which nonpersistent messages are transported on the channel (FAST or NORMAL).

XMITQ
displays the transmission queue used by this channel if CHLTYPE is SDR.

INDOUBT
displays if the channel is in an in-doubt condition (YES or NO).

CUR SEQ NUM
displays the sequence number for the last message processed (sent or received) by the channel.

LST SEQ NUM
displays the sequence number for the last message that has been acknowledged.

LST MSG TIM
displays the time when the last message was processed.

TIMESTARTED
displays the time when the channel was started.

LST MSG DAT
displays the date when the last message was processed.

DATESTARTED
displays the date when the channel was started.

CUR LUWID
displays the logical unit of work identifier (LUWID) for the last message processed by the channel.

LAST LUWID
displays the LUWID for the last message acknowledged.

MESSAGES
displays the total number of messages processed by the channel.

BATCHES
displays the total number of batches processed by the channel.

BYTES RCVD
displays the total number of bytes received from the remote channel.

BYTES SENT
displays the total number of bytes sent to the remote channel.

BUFS RCVD
displays the total number of buffers received from the remote channel.

BUFS SENT
displays the total number of buffers sent to the remote channel.

BATCHSIZE
displays the negotiated batch size currently being used on the channel.

SEQ WRAPVAL
displays the sequence wrap value for the channel.

```
User:    ZMQSC DIS CHS-TPF.TO.NT.1

System:  MQSC0085I 06.57.41 CHANNEL TPF.TO.NT.1 READY
         CHLTYPE     - SDR
         CONNAME     - 9.117.147.92
         NPMSPEED    - NORMAL
         XMITQ       - TPFXMITQ1              INDOUBT     - NO
         CUR SEQ NUM - 37                     LST SEQ NUM - 37
         LST MSG TIM - 11.57.32               TIMESTARTED - 11.57.30
         LST MSG DAT - 2002-01-16             DATESTARTED - 2002-01-16
         CUR LUWID   - B0FFF011C35E3A01       LAST LUWID  - B0FFF00F7161BC07
         MESSAGES    - 48                     BATCHES     - 14
         BYTES RCVD  - 544                    BYTES SENT  - 28045
         BUFS RCVD   - 19                     BUFS SENT   - 56
         BATCHSIZE   - 10                     SEQ WRAPVAL - 999999999
         SOCK BUFFER - Default                LOCAL IP    -
         END OF DISPLAY
```

The following example displays the local queue manager name and TPF MQSeries profile.

```
User:    ZMQSC DISPLAY QMGR

System:  MQSC0203I 08.11.25 QUEUE MANAGER PROFILE DISPLAY
         QManager Name              - TPFQMGR
         QManager Version           - 2
         Current State              - STARTED
         Default QDEPTHHI           - 25
         Q Depth Check Interval     - 15
         Default Batch Size         - 500
         Default Max Message Length - 1040000
         Default Max Queue Depth    - 500
         Default Sweep              - ON
         Checkpoint Interval        - 5
         Dead Letter Queue Name     - DEAD.LETTER.QUEU
         QUEUE MANAGER STARTED ON: B
         END OF DISPLAY
```

The following example displays information for normal usage local queue TPFLOCALQ1, where:

Current Depth
: displays the total number of messages currently on the queue.

Persistent Msgs
: displays the total number of persistent messages, along with their total length in bytes, that were placed on the queue since the queue manager was last started.

NonPersist Msgs
: displays the total number of nonpersistent messages, along with their total length in bytes, that were placed on the queue since the queue manager was last started.

Num of MQOPEN
: displays the total number of times that applications have issued MQOPEN for the queue.

Num of MQCLOSE
: displays the total number of times that applications have issued MQCLOSE for the queue.

Default Sweep
displays whether memory queues are moved (swept) to a TPF collection support (TPFCS) persistent collection when the number of messages on the queue are increasing

```
 User:   ZMQSC DIS QL-TPFLOCALQ1

System:  MQSC0201I 09.37.17 LOCAL QUEUE DISPLAY:
         Queue Name      - TPFLOCALQ1
         QManager Name   - TPFQM
         Queue Type      - LOCAL
         Usage           - NORMAL
         Delete Pending  - NO
         Common          - NO
         Max Msg Length  - 10240
         Current Depth   - 0
         Put             - ENABLED
         Get             - ENABLED
         Persistent Msgs - 0 (0 Bytes)
         NonPersist Msgs - 0 (0 Bytes)
         Num of MQOPEN   - 0
         Num of MQCLOSE  - 0
         Process Name    -
         Trigger Data    -
         Trigger Control - ON
         Trigger Type    - FIRST
         QDEPTHHI        - 0%
         MAXDEPTH        - 5000
         SWEEP           - ON
                  END OF DISPLAY
```

The following example displays information for a normal local processor shared queue:

```
 User:   ZMQSC DIS QL-TPFCOMMONQ1

System:  MQSC0201I 09.37.55 LOCAL QUEUE DISPLAY:
         Queue Name      - TPFCOMMONQ1
         QManager Name   - TPFQM
         Queue Type      - LOCAL
         Usage           - NORMAL
         Delete Pending  - NO
         Common          - YES
         Max Msg Length  - 10240
         Current Depth   - 0
         Put             - ENABLED
         Get             - ENABLED
         Persistent Msgs - 0 (0 Bytes)
         NonPersist Msgs - 0 (0 Bytes)
         Num of MQOPEN   - 0
         Num of MQCLOSE  - 0
         Process Name    -
         Trigger Data    -
         Trigger Control - OFF
         Trigger Type    - FIRST
                  END OF DISPLAY
```

The following example displays information for local transmission queue TPFXMITQ1, where:

XmitQ Override
displays the target transmission queue if the ZMQSC SWQ command has been issued for the queue.

Current Depth
displays the total number of messages currently on the queue.

Persistent Msgs
: displays the total number of persistent messages, along with their total length in bytes, that were placed on the queue since the queue manager was last started.

NonPersist Msgs
: displays the total number of nonpersistent messages, along with their total length in bytes, that were placed on the queue since the queue manager was last started.

Num of MQOPEN
: displays the total number of times that applications have issued MQOPEN for the queue.

Num of MQCLOSE
: displays the total number of times that applications have issued MQCLOSE for the queue.

Channel Name
: displays the channel used by this transmission queue to send messages.

```
User:    ZMQSC DIS QL-TPFXMITQ1

System:  MQSC0201I 09.38.35 LOCAL QUEUE DISPLAY:
         Queue Name      - TPFXMITQ1
         QManager Name   - TPFQM
         Queue Type      - LOCAL
         Usage           - TRANSMIT
         Delete Pending  - NO
         Common          - NO
         Max Msg Length  - 10240
         Current Depth   - 0
         Put             - ENABLED
         Get             - ENABLED
         Persistent Msgs - 0 (0 Bytes)
         NonPersist Msgs - 0 (0 Bytes)
         Num of MQOPEN   - 0
         Num of MQCLOSE  - 0
         QDEPTHHI        - 0%
         MAXDEPTH        - 5000
         SWEEP           - ON
         Channel Name    - CHL1
         Swing Target    -
         END OF DISPLAY
```

The following example displays information for remote queue TPFREMOTE2, where:

Remote Queue Name
: specifies the name of the queue on the remote queue.

Remote Queue Manager
: specifies the name of the remote queue manager on which the remote queue is defined.

Transmit Queue Name
: specifies the name of the transmission queue to be used for forwarding messages to the remote queue. The `MQPUT` function is either ENABLED or DISABLED for this queue.

```
User:    ZMQSC DIS QR-RQ1

System:  MQSC0200I 07.03.00 REMOTE QUEUE DISPLAY:
         Queue Name           - TPFREMOTE2
         Queue Manager Name   - TPFQM
         Remote Queue Name    - RQ1
         Remote QManager Name - RQM1
         Transmit Queue Name  - XQ1
         Put                  - ENABLED
                           END OF DISPLAY
```

The following example displays information for alias queue AQ1.

```
User:    ZMQSC DIS QA-AQ1

System:  MQSC0204I 07.01.51 ALIAS QUEUE DISPLAY:
         Queue Name         - AQ1
         QManager Name      - TPFQM
         Queue Type         - ALIAS
         Base Queue         - RQ1
         Put                - ENABLED
         Get                - ENABLED
         END OF DISPLAY
```

The following example displays information for channel TPF.TO.NT.1.

The following example displays the current status for all channels.

```
User:    ZMQSC DIS CHS-ALL

System:  MQSC0208I 07.01.51 CHANNEL STATUS DISPLAY:
         CHANNEL          CHL          CURRENT
         NAME             TYPE         STATUS
         ----------------------------------------
         CHL1             SDR          READY
         CHL2             SDR          STOPPED
         CHL3             SDR          STOPPED
         MVS.TO.TPF       RCVR         READY
         NT.TO.TPF        RCVR         STOPPED
         END OF DISPLAY
```

The following example displays the current status for all channels and indicates whether the channel is in an in-doubt condition or not.

```
User:    ZMQSC DIS CHS-ALL INDOUBT

System:  MQSC0208I 07.01.51 CHANNEL STATUS DISPLAY:
         CHANNEL          CHL          CURRENT      INDOUBT
         NAME             TYPE         STATUS       STATE
         ---------------------------------------------------
         CHL1             SDR          READY         -
         CHL2             SDR          STOPPED       N
         CHL3             SDR          STOPPED       Y
         MVS.TO.TPF       RCVR         READY         -
         NT.TO.TPF        RCVR         STOPPED       -
         END OF DISPLAY
```

The following example displays the default persistence of all the local queues.

```
User:    ZMQSC DISP QL-ALL DEFPSIST

System:  MQSC0201I 13.38.51 LOCAL QUEUE DISPLAY:
         Queue Name                    DEFPSIST
         ----------------------------- ----------
         DEAD.LETTER.QUEUE             - Not Persistent
         SPECIAL.RECOVERY.QUEUE        - Not Persistent
         SYSTEM.TEMPORARY.QUEUE        - Not Persistent
         XQ1                           - Not Persistent
         LOCAL.TPF.QUEUE               - Persistent
         XMITQ2                        - Persistent
         END OF DISPLAY
```

The following example displays the channel-type definitions of each channel.

```
User:    ZMQSC DISP CHL-ALL CHLTYPE

System:  MQSC0225I 13.48.46 CHANNEL DISPLAY FOR PROCESSOR B:
         Channel Name             CHLTYPE
         ------------------------ ------------
         NT.TO.TPF                RCVR
         TPF.TO.OS390             SDR
         TPFTONT                  SDR
         END OF DISPLAY
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC MIGRATE–Migrate TPF MQSeries Queues

Use this command to migrate existing TPF MQSeries queues to the new format established by turbo enhancements for TPF support of MQSeries local queue manager (APAR PJ27023) and to fall back TPF MQSeries queues to a TPF system prior to APAR PJ27023.

Requirements and Restrictions

- Migration must be committed before you can enter the following commands:
  - ZMQSC ALTER (QA, QL, QR, or CHL)
  - ZMQSC DEFINE (QA, QL, QR, or CHL)
  - ZMQSC DELETE
  - ZMQSC DBREBUILD.
- You cannot enter the ZMQSC MIGRATE command with the COMMIT parameter specified while there are messages on the dead-letter queue unless you also specify the FORCE parameter.
- When processor unique queues are migrated, the value for the SWEEP parameter in the queue definition is set to OFF.
- You must enter this command on each subsystem in your complex where a local queue manager was previously defined.

Format

```
►►──ZMQSC MIGrate── ──┬─BEGin──┬─────────┬────┬──────────────────────────►◄
                      │        └──FORCE──┘    │
                      ├─COMMit──┬─────────┬───┤
                      │         └──FORCE──┘   │
                      ├─FALLback──┬─────────┬─┤
                      │           └──FORCE──┘ │
                      └─STATus────────────────┘
```

BEGin
: sets the TPF MQSeries queue manager to migration mode. All processor unique queues that were defined previously will be migrated to the new format for the processor on which this command is entered. Enter **ZMQSC MIGRATE BEGIN** before starting the queue manager for the first time on a processor with turbo enhancements for TPF support of MQSeries local queue manager applied. Migration mode continues until you specify the FALLBACK or COMMIT parameter. You cannot define new queues or channels, or delete or change existing ones, while in migration mode. You must make any changes to queues or channels on a TPF system without turbo enhancements for TPF support of MQSeries local queue manager applied, or after you specify the COMMIT parameter.

COMMit
: ends the migration mode of the TPF MQSeries queue manager for all processors. Before you specify this parameter, do the following:

1. Apply turbo enhancements for TPF support of MQSeries local queue manager on all processors in the loosely coupled complex.

2. Enter **ZMQSC MIGRATE BEGIN** on all active processors in the loosely coupled complex. An active processor is one that possibly has messages on at least one processor-unique queue. A processor is marked active when you enter **ZMQSC START QMGR**. It remains active until all processor unique queues are emptied and you enter **ZMQSC STOP QMGR**. If you enter **ZMQSC START QMGR** again, the processor becomes active even if there are no messages on the processor unique queues.

FALLback
converts TPF MQSeries queues back to the format that was used before turbo enhancements for TPF support of MQSeries local queue manager. After you specify this parameter, the queue manager cannot be started on a TPF system with turbo enhancements for TPF support of MQSeries local queue manager applied.

FORCE
forces the MIGRATE BEGIN or MIGRATE FALLBACK operations to be completed or to end even if the TPF system is in a partially migrated state because of an error. This parameter also forces the MIGRATE COMMIT operation to be completed or to end even if the the old format (before APAR PJ27023) dead-letter queue is not empty.

STATus
displays the processors in a loosely coupled complex that have yet to be migrated. Any processor that has nonzero, processor unique queues in the old format must be migrated.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Processors with and without APAR PJ27023 can co-exist within a loosely coupled complex.

Examples

The following example migrates a TPF MQSeries queue.

```
User:   ZMQSC MIGRATE BEGIN

System: MQSC0173I ALL QUEUES SUCCESSFULLY MIGRATED
```

Related Information

- See *MQSeries Command Reference* for more information about the MQSeries product.
- See *TPF Migration Guide: Program Update Tapes* for more information about migrating TPF MQSeries queues to the turbo enhancements for TPF support of MQSeries local queue manager format.

ZMQSC MOVEMSGS–Move Messages

Use this command to move messages in a memory queue from a deactivated processor to another processor in the loosely coupled complex.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You can enter this command only when the local MQSeries queue manager is started.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.
- You cannot move messages from a transmission queue.
- The processor specified with the CPU parameter must be deactivated from the loosely coupled complex.

Format

```
►►──ZMQSC MOVEMSGS── ──Queue-qname── ──CPU-cpuid──┬───────────┬──────────────►◄
                                                  └── ──FORCE─┘
```

Queue-*qname*
: specifies the normal local queue name on the originating central processing unit (CPU) whose messages will be moved to the CPU that entered this command, where *qname* is a 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). A queue name cannot begin or end with a period and cannot contain two consecutive periods.

CPU-*cpuid*
: specifies the CPU identifier (ID) of the originating processor, where *cpuid* is a 1-character CPU ID.

FORCE
: forces messages to be moved.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

In the following example, messages are moved successfully.

```
User:   ZMQSC MOVEMSGS QUEUE-Q1 CPU-C

System: MQSC0220I 200 MESSAGES MOVED FROM QUEUE - Q1 SUCCESSFULLY
```

In the following example, messages are not moved successfully.

```
User:   ZMQSC MOVEMSGS QUEUE-Q2 CPU-C

System: MQSC223E 200 MESSAGES MOVED FROM QUEUE - Q2,  MOVE ABORTED,
        UNABLE TO PUT MESSAGES WITH REASON CODE - 2030
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC RESET–Reset a Sender or Receiver Channel

Use this command to reset a sender or receiver channel with a sequence number because the sequence number does not match the number of the remote message channel agent (MCA).

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
                            ┌ ──SEQNUM-1──────┐
►►──ZMQSC RESET── ──CHL-name─┼─────────────────┼──────────────────►◄
                            └ ──SEQNUM-number─┘
```

CHL-*name*
: specifies the name of the channel to be reset, where *name* is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

SEQNUM-*number*
: specifies the sequence number, where *number* is a number from 1 to the value specified for the SEQWRAP parameter of the ZMQSC DEF CHL command.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZMQSC HELP
ZMQSC ?

Examples

The following example resets a sender channel.

```
User:   ZMQSC RESET CHL-TPF.TO.JOHN SEQNUM-99999

System: MQSC0316I 12.08.32 CHANNEL - TPF.TO.JOHN HAS BEEN RESET
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC RESOLVE–Resolve a Sender Channel

Use this command to resolve a sender channel when the channel is in an in-doubt condition; that is, when the TPF sender channel does not know if the remote channel received the last batch of messages before the channel was stopped.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC RESOLVE── ──CHL-name── ──ACTION-─┬─COMMIT──┬──────────────────────────────►◄
                                           └─BACKOUT─┘
```

CHL-*name*
specifies the name of the channel to be resolved, where *name* is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

ACTION
specifies if the last batch has been sent successfully, where:

COMMIT
specifies that the batch has been sent successfully.

BACKOUT
specifies that the batch has not been sent successfully and the messages are placed back on the transmission queue.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZMQSC HELP
ZMQSC ?

Examples

The following example resolves a sender channel.

```
User:   ZMQSC RESOLVE CHL-TPF.TO.JOHN ACTION-BACKOUT

System: MQSC0320I 12.08.32 RESOLVE CHANNEL - TPF.TO.JOHN SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC START–Start a TPF MQSeries Channel or Queue Manager

Use this command to start a TPF MQSeries sender channel or queue manager, or enable a TPF MQSeries receiver channel.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.
- **Attention:** If a ZMQSC SWQ command is in progress, do not start the channel associated with the queue specified as the source transmission queue (FROMQ). If you do, messages in this queue may be transmitted out of sequence.

Format

```
►►──ZMQSC START── ──┬─CHL-─┬─name─┬─┬──────────────────────────────►◄
                    │      └─ALL──┘ │
                    └─QMGR──────────┘
```

CHL
specifies the name of the channel to be started or enabled, where:

name
is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

If you specify a sender channel, the channel is started. If you specify a receiver or server connection channel, and if the channel was previously disabled with the ZMQSC STOP command, the channel is enabled.

ALL
starts all sender channels defined to the TPF MQSeries queue manager.

QMGR
starts the TPF MQSeries queue manager.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- If the queue manager is started and the TPF system is subsequently cycled down from NORM state or an initial program load (IPL) is performed, the queue manager will be automatically started when the TPF system cycles back to NORM state.

Examples

The following example starts a TPF MQSeries sender channel.

```
User:   ZMQSC STA CHL-TPF.TO.JOHN

System: MQSC0315I 12.08.32 START SENDER CHANNEL - TPF.TO.JOHN SUCCESSFUL
```

The following example starts a TPF MQSeries queue manager.

```
User:   ZMQSC STA QMGR

System: MQSC0019I 12.08.32 START QUEUE MANAGER SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC STOP–Stop a TPF MQSeries Channel or Queue Manager

Use this command to stop a TPF MQSeries sender or receiver channel or the TPF MQSeries queue manager.

Requirements and Restrictions

- The TPF MQSeries channel or TPF MQSeries queue manager can be stopped only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
►►──ZMQSC STOP──  ──┬─CHL-──┬─name──┬──────────┬─┬─┬──────────────────────────────►◄
                    │       │       └── ──FORCE─┘ │ │
                    │       └─ALL─────────────────┘ │
                    └─QMGR──────────────────────────┘
```

CHL
: specifies the name of the channel to be stopped, where:

name
: is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

  If you specify a receiver or server connection channel, the channel is disabled. The channel cannot be started again until you enable the channel with the ZMQSC START command.

ALL
: stops all MQSeries servers and channels.

FORCE
: stops the transmission of any current batch of messages.

QMGR
: stops the TPF MQSeries queue manager.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- Only the channel or channels defined locally are stopped.
- Only the local TPF MQSeries queue manager is stopped.

Examples

The following example stops a TPF MQSeries sender channel.

```
User:   ZMQSC STOP CHL-TPF.TO.JOHN

System: MQSC0311I 12.08.32 CHANNEL TPF.TO.JOHN STOPPED
```

The following example stops a TPF MQSeries queue manager.

```
User:   ZMQSC STOP QMGR

System: MQSC0020I 12.08.32 STOP QUEUE MANAGER SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC SWQ–Swing a TPF MQSeries Transmission Queue

Use this command to move the messages from one transmission queue (a source queue) to another transmission queue (a target queue), and to have future messages sent to the target queue.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.
- Channels associated with the transmission queue must be stopped and not in an in-doubt condition.
- **Attention:** If a ZMQSC SWQ command is in progress, do not start the channel associated with the queue specified as the source transmission queue (FROMQ). If you do, messages in this queue may be transmitted out of sequence.

Format

```
►►──ZMQSC SWQ── ──FROMQ-fqname── ──TOQ-tqname──────────────────────────────►◄
```

FROMQ-*fqname*
: specifies the source transmission queue, where *fqname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9).

TOQ-*tqname*
: specifies the target transmission queue, where *tqname* is the 1- to 48-character queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?
- The queue is reset if you specify the same transmission queue name for both the FROMQ and TOQ parameters. New messages are no longer routed to a different target queue.

Examples

The following example moves the messages from transmission queue abc to transmission queue def.

```
User:   ZMQSC SWQ FROMQ-'abc' TOQ-'def'

System: MQSC0088I QUEUE SWING abc TO def SUCCESSFUL
```

Related Information

See *MQSeries Command Reference* for more information about the MQSeries product.

ZMQSC TRACE–Tracing a TPF MQSeries Channel or Queue

Use this command to do the following:

- Set or display trace parameters
- Start or stop tracing according to the current trace parameters.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You must define the queue manager profile by entering the ZMQSC DEF MQP command before entering this command.
- Without single quotation marks (' ') specified, characters are handled as uppercase only. If a TPF MQSeries object needs to be case-sensitive, enclose the object in single quotation marks.

Format

```
>>--ZMQSC TRACE-- ---------------------------------------------------------->

                                   --Comms-----
>-------CHL----name--------------+-+------------+-- --+-ON--+-------------><
   |  | |       +-ALL-+         | |             |    | +-OFF-+ |
   |  | |                        | +--Function--+    |         |
   |  | +-CHANNEL----name--------+ +--ALL--------+    |         |
   |  |             +-ALL-+                           |         |
   |  +-Queue----qname------------------------------+         |
   |  |          +-ALL-+                              |         |
   |  +-ADMin-----------------------------------------+         |
   |  +-ALL-------------------------------------------+         |
   +-CONSOLE----ON-------------------------------------------+
   |           +-OFF-+                                        |
   +-TAPE----ON----------------------------------------------+
   |        +-OFF-+                                           |
   +-START----------------------------------------------------+
   +-STOP-----------------------------------------------------+
   |            --STATUS----                                  |
   +-DISplay---+------------+----------------------------------+
               +--Queue-----+
               +--CHL-------+
               +--CHANNEL---+
               +--ALL-------+
```

CHL
specifies the name of the channel you want to trace, where:

name
is the 1- to 20-character channel name. A channel name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

ALL
sets tracing for all channels. This is independent of any individual channels for which trace has been started.

CHANNEL
specifies the name of the channel you want to trace. This is equivalent to the CHL parameter.

Comms
sets communications trace for the specified channel.

Function
sets function trace for the specified channel.

ALL
sets both function and communications trace for the specified channel.

Queue
specifies the queue for which function trace is to be set, where:

qname
is a 1- to 48-character single queue name. A queue name can contain percent signs (%), slashes (/), underscores (_), periods (.), letters in either uppercase (A–Z) or lowercase (a–z), and digits (0–9). If the name is enclosed in single quotation marks, the characters can be mixed case.

ALL
sets function trace for all queues. This setting is independent of any individual queues for which trace was started.

ADMin
sets function trace for all ZMQSC commands.

ALL
sets function trace for all system activity. Do not specify this parameter during high system activity because it can severely degrade performance. Setting this parameter does not affect the setting for channels, queues, or the ZMQSC commands.

ON
specifies one of the following:

- If you specify the CHL or CHANNEL parameter, the specified trace (function, communications, or both) is set on for the specified channel.
- If you specify the QUEUE parameter, function trace is set on for the specified queue. Tracing will occur even if the queue is not defined.
- If you specify the ADMIN parameter, function trace is set on for the ZMQSC commands.
- If you specify the ALL parameter, function trace is set on for all system activity.

Tracing does not actually start until you enter **ZMQSC TRACE START**. Tracing continues until you enter **ZMQSC TRACE STOP**.

OFF
specifies one of the following:

- If you specify the CHL or CHANNEL parameter, the specified trace (function, communications, or both) is set off for the specified channel.
- If you specify the QUEUE parameter, function trace is set off for the specified queue.
- If you specify the ADMIN parameter, function trace is set off for the ZMQSC commands.

- If you specify the ALL parameter, function trace is set off for all system activity. This setting is independent of activity specified with the QUEUE, CHL, CHANNEL, or ADMIN parameters.

CONSOLE
specifies one of the following:

ON
specifies that the trace output will be sent to the console. Specify this parameter only for test systems because it can degrade performance on a production system.

Note: If you enter **ZMQSC TRACE STOP** to stop tracing and there are still messages in the buffer, these messages will continue to display on the console until the buffer is empty.

OFF
specifies that the trace output will not be sent to the console.

TAPE
specifies one of the following:

ON
specifies that the trace output will be sent to the RTL tape. You must postprocess the output offline. See *TPF Program Development Support Reference* for an example of the job control language (JCL) you can use to postprocess the trace data.

OFF
specifies that the trace output will not be sent to the RTL tape.

START
starts tracing according to the current parameter settings.

STOP
stops tracing. The current parameter settings are retained.

DISplay
displays trace information, where:

STATUS
displays the current trace parameters.

Queue
displays a list of queues to be traced.

CHL
displays a list of channels to be traced.

CHANNEL
displays a list of channels to be traced.

ALL
displays the current trace parameter and a list of queues and channels to be traced.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZMQSC HELP
  ZMQSC ?

- The trace data is attached to the entry control block (ECB) and will be included in any system error dumps that are issued even if you specify NO for both the CONSOLE and TAPE parameters.

Examples

The following example sets a queue named LQ1 to be traced.

```
User:   ZMQSC TRACE Q-LQ1 ON

System: MQSC0030I 12.08.32 QUEUE LQ1 ADDED TO FUNCTION TRACE TABLE
```

The following example sets both function and communication trace on for all channels.

```
User:   ZMQSC TRACE CHL-ALL ALL ON

System: MQSC0047I 12.10.30 TRACE PARAMETERS ALTERED SUCCESSFULLY
```

The following example sets tracing off for the ZMQSC commands.

```
User:   ZMQSC TRACE ADMIN OFF

System: MQSC0047I 13.15.44 TRACE PARAMETERS ALTERED SUCCESSFULLY
```

The following example specifies that trace output will be sent to the console.

```
User:   ZMQSC TRACE CONSOLE-ON

System: MQSC0047I 13.15.44 TRACE PARAMETERS ALTERED SUCCESSFULLY
```

The following example starts tracing according to the current parameter settings.

```
User:   ZMQSC TRACE START

System: MQSC0048I 13.15.44 TRACE STARTED
```

The following example displays the current tracing parameters.

```
User:   ZMQSC TRACE DISPLAY STATUS

System: MQSC0177I 13.15.44 TRACE STATUS DISPLAY
        TRACE IS OFF
          TAPE - ON
          CONSOLE - OFF
        TRACING ALL - OFF
        TRACING ADMIN - ON
        TRACING ALL QUEUES - NO
        TRACING ALL CHANNELS (FUNCTION) - NO
                             (COMMS) - NO
        END OF DISPLAY
```

The following example displays a list of the queues being traced.

```
User:   ZMQSC TRACE DISPLAY QUEUE

System: MQSC0177I 13.15.44 TRACE STATUS DISPLAY
        QUEUES IN FUNCTION TRACE TABLE:
          LQ1
          LQ2
        END OF DISPLAY
```

Related Information

- See *MQSeries Command Reference* for more information about the MQSeries product.
- See *TPF Program Development Support Reference* for more information about TPF MQSeries tracing and postprocessing and an example of the trace data.

ZNACT–Alter SNA Printer Path (Alternate to Prime)

Use this command to activate a node for a 3270 synchronous data link control (SDLC) hardcopy resource again after an alternate node is assigned to it.

Requirements and Restrictions

Use the ZNALT command to send the output for the 3270 SDLC hardcopy resource to another 3270 SDLC hardcopy resource before you enter this command.

Format

```
►►──ZNACT── ──nodename──────────────────────────────────────────►◄
```

nodename
is a 1- to 17-character node name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNACT HELP
ZNACT ?

Examples

The TAF02O00 node is activated again in the following example.

```
User:   ZNACT TAF02O00

System: NACT0000I 08.24.49 REACTIVATE OUTPUT COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNALS–Display ALS Status

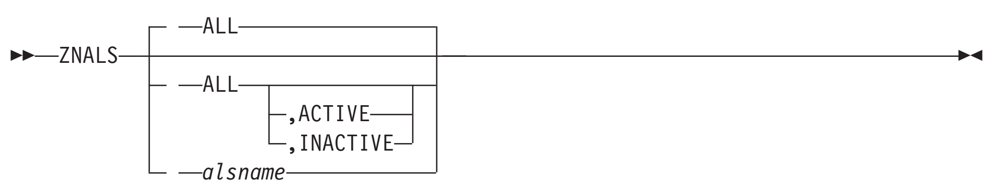
Use this command to display the current status of an adjacent link station (ALS), network control program (NCP), or channel to channel (CTC) link. The status information shows whether the ALS, NCP, or CTC link is active or inactive. If it is active, the status information also lists the following:

- Symbolic device address (SDA)
- Network ID of the ALS link
- Control logical unit (CLU) name of the ALS, if the CLU session is active
- Cross-domain resource manager (CDRM) of the CTC link
- Status of the path information unit (PIU) trace facility.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format



ALL
displays the status of each ALS.

ACTIVE
displays the status of only the active ALSs.

INACTIVE
displays the status of only the inactive ALSs.

alsname
is the 1- to 8-character name of an ALS. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNALS HELP
ZNALS ?

Examples

The following information is displayed in the examples:

NAME
is the name of the ALS

SDA
is the symbolic device address of the ALS

STATUS
is one of the following to indicate the status of the ALS:

ACTIVE/N
ALS is active, but is not polling

ACTIVE/P
ALS is active and is polling

INACTIVE
ALS in not active.

NET-ID
is the network ID of the ALS

CLU-NAME
is the CLU of the ALS, if the CLU session is active

CDRM-NAME
is the CDRM of the ALS, if the CTC link is active

TRC
is one of the following to indicate the status of the PIU trace facility:

YES
The PIU trace facility is active for the ALS.

Y/V
The PIU trace facility is active for the ALS, and only the network control commands, virtual route (VR) pacing requests, and VR pacing responses are being traced.

NO
The PIU trace facility is not active for the ALS.

The status of each ALS is displayed in the following example.

```
User:   ZNALS

System: NALS0001I 15.15.40 BEGIN ZNALS DISPLAY
          NAME    SDA    STATUS    NET-ID   CLU-NAME   CDRM-NAME   TRC
          ----    ---    ------    ------   --------   ---------   ---
        N30E521         INACTIVE
        N34H540   0865  ACTIVE/P                                   YES
        N42E431         INACTIVE
        N46E431         INACTIVE
        P34CA8B   0868  ACTIVE/P  VTAMNET                          YES
        P34CA8D         INACTIVE
        END OF ZNALS DISPLAY
```

The status of each active ALS is displayed in the following example.

```
User:   ZNALS ALL,ACTIVE

System: NALS0001I 15.18.31 BEGIN ZNALS DISPLAY
          NAME    SDA    STATUS    NET-ID   CLU-NAME   CDRM-NAME   TRC
          ----    ---    ------    ------   --------   ---------   ---
        N34H540   0865  ACTIVE/P                                   YES
        P34CA8B   0868  ACTIVE/P  VTAMNET                          YES
        END OF ZNALS DISPLAY
```

The status of the N34H540 ALS is displayed in the following example.

```
User:   ZNALS N34H540

System: NALS0001I 15.21.57 BEGIN ZNALS DISPLAY
          NAME    SDA    STATUS    NET-ID   CLU-NAME   CDRM-NAME   TRC
          ----    ---    ------    ------   --------   ---------   ---
        N34H540   0865  ACTIVE/P                                   YES
        END OF ZNALS DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNALT–Alter SNA Printer Path (Prime to Alternate)

Use this command to reroute printer output from the prime node to an alternate node.

Requirements and Restrictions

None.

Format

```
►►──ZNALT── ──primenode── ──altnode──────────────────────────────────►◄
```

primenode
: is the 1- to 17-character node name of the prime printer. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

altnode
: is the 1- to 17-character node name of the alternate printer. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNALT HELP
ZNALT ?

Examples

Printer output is rerouted from the TAF02O00 node to the TAF03O00 node in the following example.

```
User:   ZNALT TAF02O00 TAF03O00

System: NALT0001I 08.23.51 OMT Q SWITCHED FROM TAF02O00          TO TAF03O00
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNAPN–Display or Change APPN Status

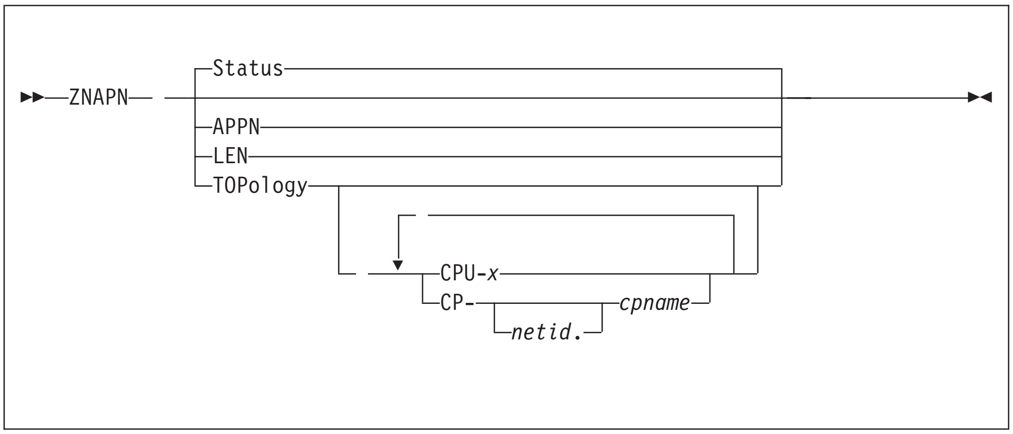
Use this command to display or change the current status of the Advanced Peer-to-Peer Networking (APPN) migration switch for the TPF system. The migration switch indicates whether the TPF system is using APPN support or low-entry networking (LEN) support.

You can also use this command to display the network topology for the TPF loosely coupled complex.

Requirements and Restrictions

You can enter the ZNAPN command only in 1052 state or higher.

Format



Status
displays the current setting of the migration switch.

APPN
sets the migration switch to APPN support.

LEN
sets the migration switch to LEN support.

TOPology
displays information about the active PU 2.1 links connected to the TPF loosely coupled complex.

CPU-*x*
displays the network topology for a specific processor in the loosely coupled complex, where *x* is the CPU ID of that processor.

CP-*netid.cpname*
displays the active links to a specific remote control point (CP), where *netid.cpname* is the 1- to 17-character name of that CP. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME.**

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNAPN HELP
ZNAPN ?

Examples

The TPF system is placed in APPN mode in the following example.

```
User:   ZNAPN APPN

System: NAPN0001I 14.39.45 APPN-LEN SWITCH SET TO APPN
```

The network topology is displayed in the following example, where:

LINK
: is the name of the adjacent link station (ALS).

CPU
: is the CPU ID of the loosely coupled processor to which the ALS is attached.

NETID
: is the network ID of the remote CP name for the node that owns the ALS.

CP NAME
: is the remote CP name for the node that owns the ALS.

TG
: is the transmission group (TG) number assigned to the link.

CP-CP
: indicates whether the link supports CP-CP sessions.

QUIESCED
: indicates whether the link has quiesced.

HPR
: indicates whether high-performance routing (HPR) support is supported on the link and, if so, whether the link connects to a rapid transport protocol (RTP) node or an automatic network routing (ANR) node.

```
User:   ZNAPN TOPOLOGY

System: NAPN0011I 15.23.15 BEGIN ZNAPN TOPOLOGY DISPLAY
          LINK    CPU  NETID   CP NAME   TG  CP-CP  QUIESCED  HPR
        --------  --- -------- -------- ---  -----  --------  ---
        P30CA1B    B  VTAMNET  VTAM2     22   YES      NO     RTP
        P30CA4A    B  VTAMNET  VTAM2     23   YES      NO     RTP
        P30CA2C    C  VTAMNET  VTAM2     21   YES      NO     RTP
        P34CA6C    C  VTAMNET  VTAM3     21    NO      NO     ANR
        END OF ZNAPN TOPOLOGY DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about:

- APPN support
- Migrating from LEN to APPN support
- HPR support.

ZNCCB–Initialize/Display CCB Information

Use this command to:

- Initialize one or all TPF Advanced Program-to-Program Communications (TPF/APPC) conversation control blocks (CCBs) or all CCBs associated with a specific LU
- Display the contents of a specific CCB
- Display a list of CCB IDs for conversations with a specific remote LU
- Display a summary of the number of active and inactive CCBs
- Display a summary of the number of CCBs in each conversation state
- Display the number of CCBs in a particular conversation state.

Requirements and Restrictions

None.

Format

```
►►──ZNCCB── ──┬─INIT── ──┬─ALL──┬────────┬──────────┬──┬──────────────────►◄
              │          │      └─,Force─┘          │  │
              │          ├─N── ─┬────────┬─nodename─┤  │
              │          │      └─netid.─┘          │  │
              │          └─O── ─ordinal─────────────┘  │
              ├─┬─DF─┬── ─O── ─ordinal─────────────────┤
              │ └─DU─┘                                 │
              └─DC── ──┬─SUM───────────────────────────┘
                       ├─ALL──┬────────────┬─────────┤
                       │      └── ─MAPPED──┘         │
                       ├─state───────────────────────┤
                       └─N── ─┬────────┬─nodename────┘
                              └─netid.─┘
```

INIT
initializes CCB resources and the fields in the session control block (SCB) that are associated with the CCB.

Note: Use this parameter only in a test environment and only when the TPF system is in NORM state. Initializing an active CCB clears the CCB entry and can cause unpredictable results.

ALL
initializes the lost CCBs. Any *lost* CCBs, which are CCBs that are marked as in use but are not actually being used, are returned to the pool of available CCBs.

Force
forces all CCBs to be initialized.

N *netid.nodename*
initializes all CCBs associated with the specified NAU, where *netid* is the 1- to 8-character network identifier and *nodename* is the 1- to 8-character LU

name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

O *ordinal*
initializes the CCB resource for the specified ordinal number, where *ordinal* is the hexadecimal ordinal number.

DF
produces a formatted display for the specified CCB resources.

O *ordinal*
displays the CCB resources for the specified ordinal number, where *ordinal* is the hexadecimal ordinal number.

DU
produces an unformatted display for the specified CCB resources.

DC
displays a summary of the number of CCBs.

SUM
displays a summary of the number of active and inactive CCBs.

ALL
displays a summary of the number of CCBs in each conversation state.

MAPPED
displays a summary of the number of CCBs in each mapped conversation state. If you omit this parameter, the basic conversation states are displayed.

state
displays the number of CCBs in a particular basic conversation state. The valid conversation state codes are:

00 RESET

01 SEND

02 RECEIVE

03 RECEIVED CONFIRM

04 RECEIVED CONFIRM SEND

05 RECEIVED CONFIRM DEALLOCATE

08 PENDING DEALLOCATE

09 END.

Note: TPF does not support the other conversation state codes defined by the LU 6.2 architecture. Conversations using the TPF mapped conversation interface are counted by their underlying basic conversation state.

N *netid.nodename*
displays the CCB IDs associated with the specified NAU, where *netid* is the 1- to 8-character network identifier, and *nodename* is the 1- to 8-character

LU name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNCCB HELP
ZNCCB ?

Examples

The formatted (ZNCCB DF) and unformatted (ZNCCB DU) displays in the examples provide the following information:

Note: Not all of the information is in both displays.

ALOCQ
: is the pointer to the next CCB entry, if any, on the queued ALLOCATE list for a given (LU,MODE) pair.

AORI
: is the ACTIVATE_ON_RECEIPT and ACTIVATE_ON_CONFIRMATION indicator. In the formatted display, one of the following values is shown (see note 1 on page 929):
- AOC DEALLOCATE PENDING
- AOC PREP TO RCV PENDING
- AOC CONFIRM PENDING
- AOR PENDING.

BTYP
: is the receive buffer type.

CCBADR
: is the address of the CCB.

CCBID
: is the conversation ID.

CEBR
: is the conditional end bracket received.

CFML
: is the confirm locks.

CONV
: is the conversation state.

CS
: is the mapped conversation state. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
- CM INITIALIZE STATE
- CM SEND STATE
- CM RECEIVE STATE

- CM CONFIRM STATE
- CM CONFIRM SEND STATE
- CM CONFIRM DEALLOCATE STATE
- INVALID.

CTYPE
is the conversation type. In the formatted display, one of the following values is shown:
- BASIC
- MAPPED.

DFCI1
is the CCB control indicator.

DT
is the mapped conversation deallocate type. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
- CM DEALLOCATE SYNC LEVEL
- CM DEALLOCATE FLUSH
- CM DEALLOCATE CONFIRM
- CM DEALLOCATE ABEND
- INVALID.

ED
is the mapped conversation error direction. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
- CM RECEIVE ERROR
- CM SEND ERROR.

ERRF
is the state of FSM_ERROR_OR_FAILURE.

EVT
is the event indicator.

FLAG
is the CCB flag control indicator.

FLAG2
is the CCB flag 2 control indicator.

IND1
is the control indicator.

INPF
is the file address of first record in the CCB inbound queue.

INPL
is the file address of last record in the CCB inbound queue.

LDAT
is the amount of data processed for the current logical record being sent.

LL is the logical length of the current record being sent.

LLREM
is the logical length of the current logical record being received.

LUNAME
is the remote logical unit name.

NET ID
: is the network ID of the remote LU.

OUTB
: is the output buffer address.

PARM
: is the 8-byte hexadecimal value of the token if PARMI indicates a token. If PARMI indicates a data level, PARM is the file address of where the block is saved. See note 1 on page 929.

PARMI
: is one of the following values in the formatted display (see note 1 on page 929):

  TOKEN
  : Indicates that the transaction program sent a TPF/APPC ACTIVATE_ON_CONFIRMATION or ACTIVATE_ON_RECEIPT verb with the TOKEN parameter.

  DATALEVEL
  : Indicates that the transaction program sent an ACTIVATE_ON_CONFIRMATION or ACTIVATE_ON_RECEIPT verb with the DL parameter.

  In the unformatted display, a value of X'FF' indicates that PARM is a token. Any other value indicates that PARM is a data level.

PBLK
: is the address of the partner block, if any, used to start a mapped conversation.

PLEN
: is the length of data needed to POST this conversation.

PMOD
: is the mode name specified on the ALLOCATE verb. See note 3 on page 929.

POST
: is the state of FSM_POST.

PRID
: is the RID of the remote LU specified on the ALLOCATE verb. See note 3 on page 929.

PROG
: is the program name specified on the ACTIVATE_ON_RECEIPT or ACTIVATE_ON_CONFIRMATION verb. See note 1 on page 929.

PTR
: is the mapped conversation prepare to receive type. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
  - CM PREP TO RECEIVE SYNC LEVEL
  - CM PREP TO RECEIVE FLUSH
  - CM PREP TO RECEIVE CONFIRM
  - INVALID.

RC
: is the mapped conversation return control type. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
  - CM WHEN SESSION ALLOCATED
  - CM IMMEDIATE

- INVALID.

RES
: is reserved for IBM use.

RFSM
: is the state of CCB. In the formatted display, one of the following values is shown:
- FREE
- INUSE
- PENDING SCB.

RID
: is the hexadecimal resource identifier (RID) of the partner logical unit.

RTSRI
: is the request to send received indicator.

RVT1ADR
: is the address of resource vector table section 1 for this session.

SCBID
: is the session control block ID.

SCB1ADR
: is the address of session control block section 1.

SL
: is the mapped conversation synchronization level. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
- CM NONE
- CM CONFIRM
- INVALID.

SLEVL
: is the synchronization level. This information is provided only in the formatted display and is one of the following values:
- CONFIRM
- NONE.

SON
: is the session outage indicator.

ST
: is the mapped conversation send type. In the formatted display, one of the following values is shown if TPF mapped conversation support is used for this conversation:
- CM BUFFER DATA
- CM SEND AND FLUSH
- CM SEND AND CONFIRM
- CM SEND AND PREP TO RECEIVE
- CM SEND AND DEALLOCATE
- INVALID.

TCBID
: is a unique ID that identifies a transaction program instance.

TEXT
: is the first 25 bytes of the first block in the CCB inbound queue. See note 2 on page 929.

TLEN
: is the text length of first record in the CCB inbound queue. See note 2.

TPDBI
: is the database index in which the transaction program is running.

TPFMAP
: indicates if this conversation is using the TPF mapped conversation support. This is provided only in the formatted display.

TPORD
: is the ordinal number of the transaction program instance control block, if any, associated with this conversation. The ITPICB data macro maps and explains the information in this block.

Notes:

1. In the formatted display, the AORI, PARM, PARMI, and PROG values are displayed only if an ACTIVATE_ON_RECEIPT or ACTIVATE_ON_CONFIRMATION is pending.
2. TEXT and TLEN are provided only in the formatted display and are displayed only if data is on the inbound queue.
3. PRID and PMOD are displayed only under all of the following conditions:
   a. The application is running in a loosely coupled environment.
   b. The local LU being used for the session is a generic TPF/APPC LU.
   c. The remote LU is already in session with another processor in the complex.

Formatted information about a CCB using basic conversation verbs is displayed in the following example.

```
User:   ZNCCB DF O 1

System: NCCB0020I 16.58.55  CCB CONTENTS FORMATTED
        CCBID    00000119  NET ID              LUNAME  APPC
        RID      000040    RVT1ADR 025D5800
        SCBID    800002    SCB1ADR 026C00F4
        CCBADR   026BC2BC  TCBID   338319C0    CTYPE   BASIC       SLEVL   CONFIRM
        CONV     SEND                          ERRF    INITIAL
        POST     INITIAL                       RFSM    INUSE
        INPF     00000000  INPL    00000000
        TPFMAP   NO
        END OF THE DISPLAY
```

Formatted information about a CCB using mapped conversation verbs is displayed in the following example.

```
User:   ZNCCB DF O 1

System: NCCB0020I 09.17.57 CCB CONTENTS FORMATTED
        CCBID    0001F72D  NET ID   LLL        LUNAME  LU62A008
        RID      000081    RVT1ADR  001F9A94
        SCBID    800007    SCB1ADR  0032F658
        CCBADR   0022BCD0  TCBID    EC46BD38   CTYPE   MAPPED    SLEVL  CONFIRM
        CONV     SEND                          ERRF    INITIAL
        POST     INITIAL                       RFSM    INUSE
        INPF     00000000  INPL     00000000
        TPFMAP   YES
           SL  CM CONFIRM                   CS  CM SEND STATE
           DT  CM DEALLOCATE SYNC LEVEL     PTR CM PREP TO RECEIVE SYNC LEVEL
           RC  CM WHEN SESSION ALLOCATED    ST  CM SEND AND PREP TO RECEIVE
           ED  CM RECEIVE ERROR
        END OF THE DISPLAY
```

Unformatted information about the CCB with ordinal number 1 is displayed in the following example.

Note: See the ICCB DSECT for information about the actual bit settings and values.

```
User:   ZNCCB DU O 1

System: NCCB0021I 17.02.26  CCB CONTENTS
        NET ID            LUNAME    APPC
        RVT1ADR  025D5800 CCBADR    026BC2BC   SCB1ADR   026C00F4
        IND1  00 02       EVT   01  00         PROG  02  00000000
        PLEN  06 7FFF     RES   08  00         SCBID 09  800002
        TCBID 0C 338319C0 CCBID 10  00000119   CTYPE 14  20
        LLREM 15 0000     DFCI1 17  00
        PARM  18 00000000 00000000             INPF  20  00000000
        INPL  24 00000000 CONV  28  01         ERRF  29  00
        POST  2A 00       RFSM  2B  01         CEBR  2C  00
        SON   2D 00       LL    2E  0000       LDAT  30  0000
        BTYP  32 00       RTSRI 33  00         CFML  34  00
        FLAG  35 00       RES   36  00         FLAG2 37  40
        OUTB  38 C00070AB PRID  3D  000000     AORI  40  00
        PARMI 41 00       TPORD 42  0000       ALOCQ 44  00000000
        PMOD  48 00000000 00000000             PBLK  50  00000000
        SL    54 00       CS    55  00         DT    56  00
        PTR   57 00       RC    58  00         ST    59  00
        ED    5B 00       TPDBI 5C  FF00
        END OF THE DISPLAY
```

All the CCBs associated with the specified NAU are initialized in the following example.

```
User:   ZNCCB INIT N LU62A003

System: NCCB0022I 17.14.26 CCB INITIALIZATION COMPLETE
```

The number of active and inactive CCBs is displayed in the following example.

```
User:   ZNCCB DC SUM

System: NCCB0023I 09.17.34 CCB SUMMARY INFORMATION
        RESOURCE STATUS
        NUMBER OF ACTIVE CCBS        50
        NUMBER OF INACTIVE CCBS    1950
        TOTAL NUMBER OF CCBS       2000
```

A summary of the number of CCBs in each basic conversation state is displayed in the following example.

```
User:   ZNCCB DC ALL

System: NCCB0024I 09:15:45 CCB CONVERSATION STATES
             CONVERSATION STATE    NUMBER    COUNT     EVENTS
        RESET                        00       1930          0
        SEND                         01         22         12
        RECEIVE                      02         16          0
        RECEIVED CONFIRM             03          4          0
        RECEIVED CONFIRM SEND        04          6          0
        RECEIVED CONFIRM DEALLOCATE  05          8          0
        PENDING DEALLOCATE           08         12          4
        END                          09          2          0
                                          --------   --------
        TOTALS                                2000         16
```

A summary of the number of CCBs in each mapped conversation state is displayed in the following example.

```
User:   ZNCCB DC ALL MAPPED

System: NCCB0028I 09:15:45 ACTIVE TPF MAPPED CONVERSATIONS
             CONVERSATION STATE    NUMBER    COUNT
        CM INITIALIZE STATE          02         2
        CM SEND STATE                03        12
        CM RECEIVE STATE             04        11
        CM SEND PENDING STATE        05         2
        CM CONFIRM STATE             06         3
        CM CONFIRM SEND STATE        07         1
        CM CONFIRM DEALLOCATE STATE  08         6
                                          --------
        TOTAL                                  37
```

The CCB IDs associated with the specified NAU are displayed in the following example.

```
User:   ZNCCB DC N LU62A008

System: NCCB0026I 09.18.45  CCBID LIST FOR LU62A008
        0000014D 00000277 00012409 00000533 00035603 00046190
        00001777 00005645
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNCNS CHANGE–Change Session Limit

Use this command to change the TPF Advanced Program-to-Program Communications (TPF/APPC) session limit from one nonzero value to another. You can also change the contention-winner polarities for parallel session connections.

Note: Throughout the description of this command, the phrase *source LU* refers to the TPF LU and *target LU* refers to the remote LU, unless otherwise noted.

Requirements and Restrictions

- You can use this command only for parallel sessions.
- You cannot use this command for the SNA-defined mode name SNASVCMG or for modes initialized as single sessions.
- The TPF system must be in CRAS state or higher.

Format

```
►►──ZNCNS Change── ──LUname-┬────────┬luname── ──Mode-modename──────────────►
                            └netid.──┘

  ┌─ ──LImit-2───────┐ ┌─ ──CONW-0───────┐ ┌─ ──CONL-0───────┐
►─┼──────────────────┼─┼─────────────────┼─┼─────────────────┼──────────────►
  └─ ──LImit-totsess─┘ └─ ──CONW-winners─┘ └─ ──CONL-losers──┘

  ┌─ ──Resp-SOURCE─┐
►─┼────────────────┼────────────────────────────────────────────────────────►◄
  └─ ──Resp-TARGET─┘
```

LUname-*netid.luname*
: is the 1- to 17-character name of the remote partner LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Mode-*modename*
: is the 1- to 8-character mode name for which the session limit and polarities are to be changed. Mode names must begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Note: To specify a mode name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the name. For example, if the mode name is $MODNAME, you must type **$$MODNAME**.

LImit-*totsess*
: specifies the maximum number of sessions that are allowed between the source LU and the target LU for the specified mode name, where *totsess* is a

decimal number greater than 0 and greater than or equal to the sum of the numbers specified for the CONW and CONL parameters.

CONW-*winners*
specifies the minimum number of sessions for which the source LU is designated to be the contention winner, where *winners* is a decimal number greater than or equal to 0.

Note: The sum of this parameter and the CONL parameter cannot exceed the value specified for the LIMIT parameter.

CONL-*losers*
specifies the minimum number of sessions for which the source LU is designated to be the contention loser, where *losers* is a decimal number greater than or equal to 0.

Note: The sum of this parameter and the CONW parameter cannot exceed the value specified for the LIMIT parameter.

Resp
specifies the LU that is responsible for selecting and deactivating sessions when the session limit decreases or the maximum number of contention winners for the source or target LUs cannot be reached, where:

SOURCE
specifies that the source LU is responsible for selecting and deactivating sessions. The target LU cannot negotiate this argument.

TARGET
specifies that the target LU is responsible for selecting and deactivating sessions. The target LU can negotiate this argument to make the source LU responsible.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNCNS HELP
  ZNCNS ?
- The LIMIT, CONW, and CONL parameters are negotiable. When the target LU chooses to negotiate these parameters, the new negotiated numbers become the new LIMIT, CONW, and CONL values.

Examples

The session limits are changed by both a TPF user and a remote user in the following examples.

In the following example, a request is made from the TPF system to change the session limit and the contention-winner polarities.

The new session limit is 4. The minimum number of contention-winner sessions for the TPF system is 4. The minimum number of contention-loser sessions for the TPF system is 0. The target (remote) LU is responsible for selecting and deactivating sessions if necessary.

```
User:   ZNCNS CH LU-APPC MODE-HATESNOW LIMIT-4 CONW-4 CONL-0 RESP-TARGET

System: NCNS0003I 12.48.47 REQUEST ACCEPTED FOR LU-APPC MODE-HATESNOW
        NCNS0001I 12.48.47 REQUEST COMPLETED FOR LU-APPC MODE-HATESNOW
```

In the following example, a request is made from the remote system to change the session limit and the contention-winner polarities. The command is entered from the remote system and an informational message is displayed on the TPF system.

The new session limit is 4. The minimum number of contention-winner sessions *for the remote side* is 4. (The number of contention-loser sessions for the TPF system is 4.) The minimum number of contention-loser sessions *for the remote side* is 0. (The number of contention-winner sessions for the TPF system is 0.) The target (TPF) LU is responsible for selecting and deactivating sessions if necessary.

Note: Message CHNX0001I displays only on the RO; it does not display on the prime CRAS.

```
User:   ZNCNS CH LU-APPA MODE-HATESNOW LIMIT-4 CONW-4 CONL-0 RESP-TARGET

System: CHNX0001I 14.02.16 CNOS CHANGE RECEIVED, MODENAME - HATESNOW,
         REMOTE LU - APPC, LOCAL LU - APPA,
         SESSION LIMIT - 4, MIN CONWINNER - 0, MIN CONLOSER - 4
```

In the following example, a request is made from the TPF system to change the session limit and the contention-winner polarities.

The new session limit is 300. The minimum number of contention-winner sessions for the TPF system is 100. The minimum number of contention-loser sessions for the TPF system is 50. The source (TPF) LU is responsible for selecting and deactivating sessions if necessary.

```
User:   ZNCNS C LU-APPC  MODE-SAM  LIMIT-300  CONW-100  CONL-50

System: NCNS0003I 12.53.18 REQUEST ACCEPTED FOR LU-APPC MODE-SAM
        NCNS0001I 12.53.18 REQUEST COMPLETED FOR LU-APPC MODE-SAM
```

In the following example, a request is made from the remote system to change the session limit and the contention-winner polarities. In this example, the command is entered from the remote system and an informational message is displayed on the TPF system.

The new session limit is 300. The minimum number of contention-winner sessions *for the remote side* is 100. (The number of contention-loser sessions for the TPF system is 100.) The minimum number of contention-loser sessions is 50. (The number of contention-winner sessions for the TPF system is 50.) The source (remote) LU is responsible for selecting and deactivating sessions if necessary.

```
User:   ZNCNS C LU-TPFNET.APPA  MODE-SAM  LIMIT-300  CONW-100  CONL-50

System: CHNX0001I 12.53.20 CNOS CHANGE RECEIVED, MODENAME - SAM,
         REMOTE LU - APPC, LOCAL - APPA,
         SESSION LIMIT - 300, MIN CONWINNER - 50, MIN CONLOSER - 100
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNCNS INITIALIZE–Initialize Session Limit

Use this command to initialize the TPF Advanced Program-to-Program Communications (TPF/APPC) session limit and contention-winner polarities for single or parallel session connections.

Note: Throughout the description of this command, the phrase *source LU* refers to the TPF LU and *target LU* refers to the remote LU, unless otherwise noted.

Requirements and Restrictions

The TPF system must be in CRAS state or higher.

Format

```
►►──ZNCNS Initialize── ──LUname-─┬──────────┬─luname── ──Mode-─┬─modename─┬──────►
                                 └─netid.─┘                    ├─singmode─┤
                                                               └─SNASVCMG─┘

   ┌─ ──LImit-2────────┐                            ┌─ ──CONW-0────────┐
►──┴─ ──LImit-totsess──┴─┬────────────────────┬─────┴─ ──CONW-winners──┴──────────►
                         └─ ──LOcal-luname──┘

   ┌─ ──CONL-0──────┐
►──┴─ ──CONL-losers─┴─┬───────────────────────────────────┬───────────────────►◄
                      ├─ ──CDRM-─┬──────────┬─cdrmname─┤
                      │          └─netid.─┘              │
                      └─ ──CP-─┬──────────┬─cpname───────┘
                               └─netid.─┘
```

LUname-*netid.luname*
: is the 1- to 17-character name of the remote partner LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Mode
: specifies the mode name for which the session limit and polarities are to be initialized, which can be one of the following:

  modename
  : is a 1- to 8-character user-defined parallel session mode name. Mode names must begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

    Note: To specify a mode name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the name. For example, if the mode name is $MODNAME, you must type **$$MODNAME**.

singmode
is the mode name for single sessions as defined by the SINGMODE parameter of the SNAKEY macro. Enter **ZNKEY SINGMODE** to display the mode name.

SNASVCMG
specifies the SNA-defined mode name used to exchange CNOS requests between the source and target LUs that are connected by parallel sessions.

LImit-*totsess*
specifies the maximum number of sessions that are allowed between the source LU and the target LU for the specified mode name, where *totsess* is a decimal number greater than 0 and greater than or equal to the sum of the numbers specified for the CONW and CONL parameters.

Notes:

1. When you specify MODE-*singmode*, do not specify the LIMIT parameter; the value is set to 1.
2. When you specify MODE-SNASVCMG, do not specify the LIMIT parameter; the value is set to 2.

LOcal-*luname*
specifies the 1- to 8-character local TPF/APPC LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you are using SLU threads, you must specify the LOCAL parameter for each initialization, specifying different thread names each time.
2. If you are not using secondary LU (SLU) threads, use the LOCAL parameter only when the first mode is initialized for a particular remote LU. If you omit the LOCAL parameter, the TPF system uses the default local TPF/APPC LU, which is defined using the MSGRTA macro. For subsequent initializations of this LU, the same local TPF/APPC LU is used regardless of the value specified for this parameter. After all modes are reset for this remote LU, a different local TPF/APPC LU can be used.
3. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CONW-*winners*
specifies the minimum number of sessions for which the source LU is designated to be the contention winner, where *winners* is a decimal number greater than or equal to 0.

Notes:

1. The sum of this parameter and the CONL parameter cannot exceed the value specified for the LIMIT parameter.
2. When you specify MODE-*singmode*, do not specify the CONW parameter; the value is set to 0.
3. When you specify MODE-SNASVCMG, do not specify the CONW parameter; the value is set to 1.

CONL-*losers*
specifies the minimum number of sessions for which the source LU is designated to be the contention loser, where *losers* is a decimal number greater than or equal to 0.

Notes:

1. The sum of this parameter and the CONW parameter cannot exceed the value specified for the LIMIT parameter.
2. When you specify MODE-*singmode*, do not specify the CONL parameter; the value is set to 0.
3. When you specify MODE-SNASVCMG, do not specify the CONL parameter; the value is set to 1.

CDRM-*netid.cdrmname*
specifies the 1- to 17-character name of the remote cross-domain resource manager (CDRM) that owns the specified remote LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CP-*netid.cpname*
specifies the 1- to 17-character control point (CP) name of the adjacent APPN node where the remote LU resides. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNCNS HELP
  ZNCNS ?
- When you specify MODE-*singmode*, the LIMIT, CONW, and CONL parameters are set to 1, 0, and 0, respectively. No CNOS exchange is performed with the remote LU.
- When you specify MODE-SNASVCMG, the LIMIT, CONW, and CONL parameters are set to 2, 1, and 1, respectively. No CNOS exchange is performed with the remote LU.
- When a CNOS exchange is performed with the remote LU, the LIMIT, CONW, and CONL parameters are negotiable. When the target LU chooses to negotiate these parameters, the new negotiated numbers become the new LIMIT, CONW, and CONL values.
- If you do not specify the CDRM parameter or the CP parameter, how the TPF system starts the session with the specified remote LU is determined by the following:
  – Whether the TPF system is connected to the network as a PU 5 node, PU 2.1 node, or both.
  – How sessions with the specified remote LU were activated previously.

  See *TPF ACF/SNA Data Communications Reference* for more information about how the TPF system selects the path to use when starting a new LU-LU session.

Examples

The session limits are initialized by both a TPF user and a remote user in the following examples.

The session limit and the contention-winner polarities are initialized from the TPF system in the following example.

The session limit is 2. The minimum number of contention-winner sessions for the TPF system is 1. The minimum number of contention-loser sessions for the TPF system is 1.

```
User:   ZNCNS INIT LU-VTAMNET1.APPA MODE-SNASVCMG

System: NCNS0003I 13.07.09 REQUEST ACCEPTED FOR LU-VTAMNET1.APPA MODE-SNASVCMG
        NCNS0001I 13.07.09 REQUEST COMPLETED FOR LU-VTAMNET1.APPA MODE-SNASVCMG
```

The session limit and the contention-winner polarities are initialized from the TPF system in the following example.

The session limit is 255. The minimum number of contention-winner sessions for the TPF system is 250. The minimum number of contention-loser sessions for the TPF system is 5.

```
User:   ZNCNS I LU-SVCA  LOCAL-SVCB  MODE-SAM  LIMIT-255  CONW-250  CONL-5

System: NCNS0003I 13.29.16 REQUEST ACCEPTED FOR LU-SVCA MODE-SAM
        NCNS0001I 13.29.17 REQUEST COMPLETED FOR LU-SVCA MODE-SAM
```

The session limit and the contention-winner polarities are initialized from the remote system in the following example. In this example, the command is entered from the remote system and an informational message is displayed on the TPF system.

The session limit is 255. The minimum number of contention-winner sessions *for the remote side* is 250. (The number of contention-loser sessions for the TPF system is 250.) The minimum number of contention-loser sessions *for the remote side* is 5. (The number of contention-winner sessions for the TPF system is 5.)

```
User:   ZNCNS I LU-APPA  LOCAL-APPC  MODE-SAM  LIMIT-255  CONW-250  CONL-5

System: CHNX0001I 13.29.19 CNOS INITIALIZE RECEIVED, MODENAME - SAM,
         REMOTE LU - APPC, LOCAL LU - APPA,
         SESSION LIMIT - 255, MIN CONWINNER - 5, MIN CONLOSER - 250
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNCNS RESET–Reset Session Limit

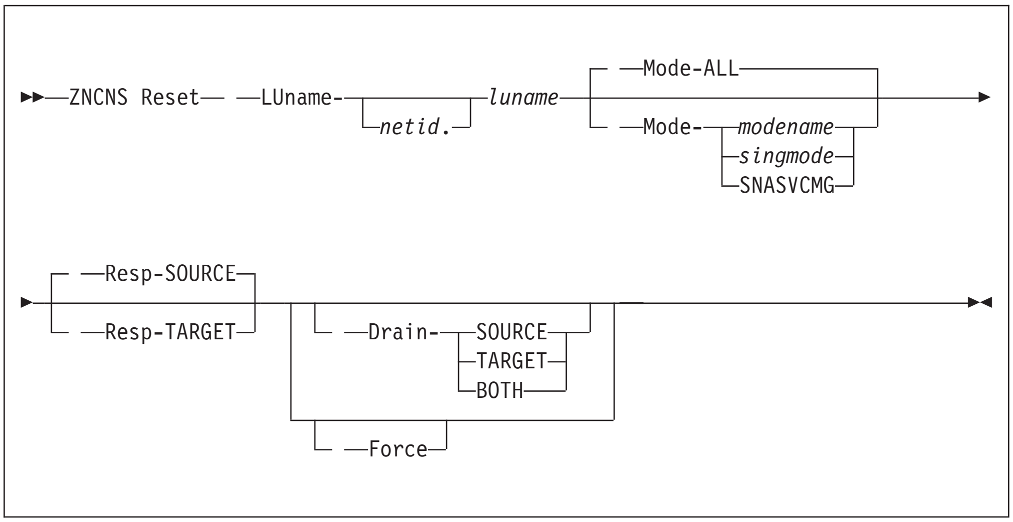
Use this command to reset the TPF Advanced Program-to-Program Communications (TPF/APPC) session limit for single or parallel session connections to 0.

Note: Throughout the description of this command, the phrase *source LU* refers to the TPF LU and *target LU* refers to the remote LU, unless otherwise noted.

Requirements and Restrictions

The TPF system must be in CRAS state or higher.

Format



LUname-*netid.luname*
: is the 1- to 17-character name of the remote partner LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Mode
: specifies the mode name for which the session limit and polarities are to be reset, which can be one of the following:

ALL
: resets to 0 the session limit for all mode names that apply to the target LU, except for the SNA-defined mode, SNASVCMG, which remains unchanged.

modename
: resets the specified 1- to 8-character user-defined mode name to 0. Mode names must begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify a mode name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the name. For example, if the mode name is $MODNAME, you must type **$$MODNAME**.

singmode
resets to 0 the specified single session mode name, which is defined by the SINGMODE parameter of the SNAKEY macro. Enter **ZNKEY SINGMODE** to display the mode name.

SNASVCMG
resets to 0 the SNA-defined mode name, which is used to exchange CNOS requests between the source and target LUs connected by parallel sessions.

Resp
specifies the LU that is responsible for deactivating sessions when the session limit for parallel session connections is reset.

Note: This parameter is not valid for single session connections or the SNASVCMG session.

SOURCE
specifies that the source LU is responsible for deactivating sessions. The target LU cannot negotiate this argument.

TARGET
specifies that the target LU is responsible for deactivating sessions. The target LU can negotiate this argument to make the source LU responsible.

Drain
specifies whether the source or target LU can drain its allocation requests. If you do not specify the DRAIN parameter, draining is not performed and all outstanding and subsequent allocation requests are rejected.

Note: This parameter is not valid for SNASVCMG connections.

SOURCE
specifies that the source LU can drain its allocation requests. The target LU cannot negotiate this argument. The source LU continues to allocate conversations to the sessions until no requests are awaiting allocation, at which time its draining has ended. Allocation requests subsequent to the completion of the CNOS exchange are rejected. Allocation requests that were queued before the CNOS RESET request are allowed to drain.

TARGET
specifies that the target LU can drain its allocation requests. The target LU can reject this argument; in this case no draining takes place. The target LU continues to allocate conversations to the sessions until no requests are awaiting allocation, at which time its draining ends. Allocation requests subsequent to the completion of the CNOS exchange are rejected. Allocation requests that were queued before to the CNOS RESET request are allowed to drain.

Note: This value is not valid for the single session connections.

BOTH
specifies that both the source LU and the target LU can drain their allocation requests.

Note: This value is not valid for the single session connections.

Force
resets the session limit of the source LU by force if the CNOS exchange is not completed successfully. If you performed a forced reset, the RESP parameter defaults to SOURCE and no draining by either the source or target LU takes place. If you do not specify the FORCE parameter, the session limit is reset when the CNOS exchange is completed successfully.

Note: This parameter is not valid for SNASVCMG or single session connections.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNCNS HELP
  ZNCNS ?
- This function in no way forces the deallocation of active conversations.

Examples

The session limits are reset by both a TPF user and a remote user in the following examples.

The session limit is reset from the TPF system in the following example. Both LUs are allowed to drain their queued allocation requests. The target (remote) LU is responsible for deactivating the sessions.

```
User:   ZNCNS RE LU-APPC MODE-ICEPICK DRAIN-BOTH RESP-TARGET

System: NCNS0003I 13.43.07 REQUEST ACCEPTED FOR LU-APPC MODE-ICEPICK
        NCNS0001I 13.43.07 REQUEST COMPLETED FOR LU-APPC MODE-ICEPICK
```

In the following example, the session limit is reset from the remote system. In this example, the command is entered from the remote system and an informational message is displayed on the TPF system. Both LUs are allowed to drain their queued allocation requests. The target (TPF) LU is responsible for deactivating the sessions.

```
User:   ZNCNS RE LU-APPA MODE-ICEPICK DRAIN-BOTH RESP-TARGET

System: CHNX0001I 13.43.09 CNOS RESET RECEIVED, MODENAME - ICEPICK
         REMOTE LU - APPC, LOCAL LU - APPA
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNCVT–Convert SNA Resource Addresses

Use this command to convert one form of an SNA address or identification to its other forms. The ZNCVT command accepts the following identification forms:

- Network address (NA)
- Logical end-point identifier (LEID)
- Resource identifier (RID) or session control block identifier (SCBID)
- Procedure correlation identifier (PCID)
- Network name
- Session address (SA)
- Session identifier (SID).

Requirements and Restrictions

None.

Format

```
►►──ZNCVT── ──┬─A── ──address──────────────┬──────────────────────────►◄
              ├─P── ──leid─────────────────┤
              ├─R── ──resource─────────────┤
              ├─D── ──┬────────────────┬─pcid─┤
              │       └─netid.cpname.──┘      │
              ├─N── ──┬─────────┬─name─────────┤
              │       └─netid.──┘             │
              ├─S── ──alsname.sesid───────────┤
              └─H── ──rtpcb.sa────────────────┘
```

A *address*
: converts a network address, where *address* is the 6-digit hexadecimal network address, which includes 2 digits for the subarea and 4 digits for the element address.

  Note: The A parameter is valid only for resources in a PU 5 environment.

P *leid*
: converts a logical end-point identifier (LEID), where *leid* is the 6-digit hexadecimal LEID.

R *resource*
: converts a resource identifier (RID) or session control block identifier (SCBID), where *resource* is the 1- to 6-digit hexadecimal RID or SCBID.

D *netid.cpname.pcid*
: converts a session services procedure correlation identifier (PCID), where *netid* is the 1- to 8-character network identifier, *cpname* is the 1- to 8-character control point name, and *pcid* is the 16-digit hexadecimal PCID. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Notes:

  1. The PCID is fully qualified when you specify the network identifier and control point name. If you do not specify the network identifier and control point name, the first entry found in the resource vector table (RVT) or session control block (SCB) with the specified PCID is displayed.

2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

N *netid.name*
converts a resource name, where *netid* is the 1- to 8-character network ID and *name* is the 1- to 8-character resource name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you do not specify the network ID, the first entry found in the RVT with the specified resource name is displayed.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

S *alsname.sesid*
converts a session identifier, where *alsname* is the 1- to 8-character ALS name and *sesid* is the 5-digit hexadecimal session ID (SID). The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. The first digit of the session ID, which represents the ODAI bit, must be 0 or 1. The 2nd–5th digits represent the session ID.
2. The S parameter is valid only for resources in session under PU 2.1 environments.
3. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME.**

H *rtpcb.sa*
converts a session address (SA), where *rtpcb* is the 1- to 6-digit hexadecimal rapid transport protocol control block (RTPCB) index and *sa* is the 16-digit hexadecimal SA. You can specify the local-assigned SA (SA1) or the remote-assigned SA (SA2).

Note: The H parameter is valid only for resources in session in a high-performance routing (HPR) network.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNCVT HELP
ZNCVT ?

Examples

A network address is converted in the following example.

```
User:   ZNCVT A 0A0015

System: NCVT0001I 13.51.16 BEGIN ZNCVT DISPLAY
         NAME - APPA                LEID - 000000
         PCID -
         RID  - 00003F
        NADDR - 0A0015
        END OF ZNCVT DISPLAY
```

A logical end-point identifier (LEID) is converted in the following example.

```
User:   ZNCVT P FE0302

System: NCVT0001I 15.50.01 BEGIN ZNCVT DISPLAY
         NAME - .LCL8DD             LEID - FE0302
         PCID -
         RID  - 00014F
        NADDR - 000000
        END OF ZNCVT DISPLAY
```

A RID is converted in the following example.

```
User:   ZNCVT R 71

System: NCVT0001I 15.50.46 BEGIN ZNCVT DISPLAY
         NAME - VTAMNET.CLUBB002     LEID - 000000
         PCID - VTAMNET.VTAM2.E383BE955248DD76
          RID - 000071
        SESID - P34CA5B.10101
        END OF ZNCVT DISPLAY
```

A resource name is converted in the following example.

```
User:   ZNCVT N APPA

System: NCVT0001I 14.44.24 BEGIN ZNCVT DISPLAY
         NAME - APPA                LEID - 000000
         PCID -
         RID  - 00003F
        NADDR - 0A0015
        END OF ZNCVT DISPLAY
```

The remote-assigned session address (SA2) is converted in the following example.

```
User:   ZNCVT H 2.2FC35E04E1E28C04

System: NCVT0007I 17.41.27 BEGIN ZNCVT DISPLAY
         NAME - VTAMNET.X604        LEID - 000000
         PCID - VTAMNET.A604.C3F6C6A90CD06438
        SCBID - 800004
          RTP - 000002  SA1 - 2FC25E04 E9162805
                        SA2 - 2FC35E04 E1E28C04
        END OF ZNCVT DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNDLU–Display LU Information

Use this command to display information about one or more logical units (LUs).

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- Information about parallel sessions is not included in displays that contain more than one LU.

Format

```
           ┌─────────────────────────────────────────────────┐
           ▼                                                 │
►►──ZNDLU──┴─┬─Id─┬─────────┬─luname─┬─────────────────┬──┬─┴────────────►◄
             │    └─netid.──┘        └──Mode-modename──┘  │
             ├─ALs-alsname────────────────────────────────┤
             ├─APplication─┬─applname─┬───────────────────┤
             │             └─NS───────┘                   │
             ├─Procedure─┬─Bind───┬───────────────────────┤
             │           └─Unbind─┘                       │
             ├─Rtp-index──────────────────────────────────┤
             └─Type─┬─0───┬───────────────────────────────┘
                    ├─1───┤
                    ├─2───┤
                    ├─3───┤
                    ├─6.2─┤
                    ├─Any─┤
                    ├─CP──┤
                    └─PLu─┘
```

ID-*netid.luname*
displays information about one or more LU, where *netid* is the 1- to 8-character network identifier and *luname* is the 1- to 8-character name of the LU. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. Use the asterisk (*) as a global file-name character to specify a group of network identifiers or LU names that begin or end with a common string of characters. For example, to display information about all the LUs that have a name beginning with PP, enter:

   ZNDLU ID-PP*

   To display information about all the LUs in the ALPHA network, enter:

   ZNDLU ID-ALPHA.*
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Mode-*modename*
displays information about the sessions started for TPF Advanced Program-to-Program Communications (TPF/APPC) LUs that have the specified mode, where *modename* is the 1- to 8-character name of the mode. Mode names must begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify a mode name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the name. For example, if the mode name is $MODNAME, you must type **$$MODNAME**.

ALs-*alsname*
displays information about the logical units (LUs) that started a session through a specific adjacent link station (ALS), network control program (NCP), or channel-to-channel (CTC) resource, where *alsname* is the 1- to 8-character name of the ALS, NCP, or CTC resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

APplication
displays information about the LUs that are in session with a specific TPF application or about the LUs that are not currently in session with a TPF application, where:

applname
is the 1- to 8-character alphanumeric name of an application.

NS
displays information about the LUs that are not currently in session with a TPF application.

Procedure
displays information about the LUs for which a session is currently being started or ended, where:

Bind
displays information about the LUs for which a session is currently being started.

Unbind
displays information about the LUs for which a session is currently being ended.

Rtp-*index*
displays information about the logical units (LUs) that are in session across a specific rapid transport protocol (RTP) connection, where *index* is the 1- to 6-digit hexadecimal rapid transport protocol control block (RTPCB) index.

Type
display information about the LUs of a particular type, where

0, 1, 2, 3, or 6.2
displays information about the LUs of type 0, 1, 2, 3, or 6.2, respectively.

Any
displays information about the LUs that are defined in the offline SNA table generation program (OSTG) as LUTYPE=ANY and are not currently in session.

Note: If an LU is defined as LUTYPE=ANY and is currently in session, that LU is considered to be of type 0, 1, 2, 3, or 6.2, depending on how the session was started.

PLu
displays information about the primary logical units (PLUs) that are not of type 6.2.

CP
displays information about the APPN control point (CP) LUs.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNDLU HELP
  ZNDLU ?
- Enter **ZNDLU** (without any parameters) to display information about every LU.
- When you enter this command, the TPF system searches the resource vector table (RVT), entry by entry, for the LUs that match the information you specified. Only the first 255 matching LUs found in the RVT are in alphabetic order by LU name in the information that is displayed. The remaining matching LUs are displayed in the order that they were found in the RVT.
- You can specify the parameters for the ZNDLU command in any order.
- If you specify an LU name and the LU name does not contain any asterisks(*), all of the other parameters that you specify in the ZNDLU command, except the MODE parameter, are ignored.

Examples

The following information is displayed in the examples:

APPL
is the name of the TPF application with which the LU is in session, or the value NS if the LU is not currently in session with a TPF application.

CDRM
is the name of the system services control program (SSCP) or the cross-domain resource manager (CDRM) that owns the LU.

Note: If no sessions were started for the LU, or if a low entry networking (LEN) session was started for the LU, this column is blank.

CLASS
Class associated with the LU, where:

APSLU
Application secondary LU

AX001
AX.25 single terminal controller

AX002
AX.25 multiple terminals controller

FMMR
: Function management message router LU

FTPI
: X.25 fast transaction processing LU

NEF
: Network extension facility LU

MCHLU
: X.25 multi-channel or physical link

VCLU
: X.25 virtual circuit LU

XALCI
: Airlines line control interconnection (ALCI) using an X.25 fast transaction processing (FTPI) LU

3600
: Finance controller

3614
: Consumer transaction facility terminal

3270C
: 3270 display device

3270P
: 3270 printer.

D is the status of the LU, where Y indicates that the LU is a dynamic LU.

LINK
: is the name of the link through which the LU has a session.

  Note: If the session is through an RTP connection that is performing a path switch, this field will be blank.

LUNAME
: is the name of the LU.

MODENAME
: is the name of the mode associated with the TPF/APPC LU session.

NETID
: is the network identifier of the LU.

  Note: If the LU is defined as NETID=ANY in the offline SNA table generation program (OSTG) and is not currently in session, this column is blank.

P is the contention status of the session, where:

- W Contention winner
- L Contention loser
- N Not currently in session.

PCID
: is the procedure correlation identifier associated with the LU.

PR
: is the procedure being performed, where:

- B A session is currently being started for the LU

U A session is currently being ended for the LU.

PROC
is the procedure being performed, where:

BIND
A session is currently being started for the LU

UNBIND
A session is currently being ended for the LU.

SCBID
is the session control block identifier.

STA
is the status of the LU, where:

A LU is active and the PIU trace facility is not active for the LU

A/P
LU is active and the PIU trace facility is active for the LU

I LU is not active and the PIU trace facility is not active for the LU

I/P
LU is not active and the PIU trace facility is active for the LU.

STATUS
is the status of the LU, where:

ACT
LU is active and the PIU trace facility is not active for the LU

ACT/P
LU is active and the PIU trace facility is active for the LU

INACT
LU is not active and the PIU trace facility is not active for the LU

INACT/P
LU is not active and the PIU trace facility is active for the LU.

TYP or TYPE
is the type associated with the LU, where:

0 Type 0

1 3270/3270E secondary logical unit (SLU) or X.25 LU

2 3270/3270E display device

3 3270/3270E SLU 3 data stream

6.2
Type 6.2

ANY
Defined in the OSTG as LUTYPE=ANY and is not currently in session

CP
APPN control point (CP) LU

PLU
Primary LU (PLU) that is not of type 6.2.

Information about all the LUs in the TPF system is displayed in the following example.

```
User:   ZNDLU

System: NDLU0031I 16.53.48 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
                .ACPAFMMR    0  I/P FMMR     NS       N/A
                .APPA       6.2 I/P
                .APPC       6.2 A/P
                .ATSO1      PLU I/P
                .ATSO3      PLU I/P
                .AZAZ       6.2 I/P
                .AZAZZ001   6.2 I/P
                .AZAZZ002   6.2 I/P
                .B32O       PLU I/P
                .CBM1       PLU I/P
                .CBM2       PLU I/P
                .CICSESA    6.2 I/P
                .CICSESA2   6.2 I/P
                .CLGB       PLU I/P
                .CLGZ       PLU I/P
                .CLG0       PLU I/P
                .CLUBB001    0  I/P CLU      NS       N/A
        VTAMNET .CLUBB002    0  A/P CLU      ELMNGR   N/A      P34CA5B
                .CLUBB003    0  I/P CLU      NS       N/A
                .CLUCC001    0  I/P CLU      NS       N/A
                .CLUC02      0  I/P CLU      NS       N/A
                .CLUC03      0  I/P CLU      NS       N/A
                .CLUC04      0  I/P CLU      NS       N/A
                .CLUC06      0  I/P CLU      NS       N/A
        END OF ZNDLU DISPLAY
```

Information about a specific LU is displayed in the following example.

```
User:   ZNDLU ID-VTAMNET.CLUBB002

System: NDLU0031I 17.05.59 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
        VTAMNET .CLUBB002    0  A/P CLU      ELMNGR   N/A      P34CA5B
        NDLU0013I 17.05.59   LU - VTAMNET.CLUBB002
        PCID - VTAMNET .VTAM2 . E383BE95 5248DD76
        END OF ZNDLU DISPLAY
```

Information about all of the LUs that have a session with the XCXC application is displayed in the following example.

```
User:   ZNDLU APPL-XCXC

System: NDLU0031I 17.08.08 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
                .PBPBB001   6.2 A/P          XCXC     TPFB     TBCTCL1
                .PBPBB002   6.2 A/P          XCXC     TPFB     TBCTCL1
                .PBPBB003   6.2 A/P          XCXC     TPFB     TBCTCL1
                .PBPBB004   6.2 A/P          XCXC     TPFB     TBCTCL1
        END OF ZNDLU DISPLAY
```

Information about all of the LUs that have a name beginning with the characters AP is displayed in the following example. Notice that information about parallel sessions is not included because the display contains more than one LU.

```
User:   ZNDLU ID-AP*

System: NDLU0031I 08.51.08 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
                .APPA       6.2 A
                .APPC       6.2 A
        END OF ZNDLU DISPLAY
```

Information about all the local and remote LU 6.2 LUs is displayed in the following example.

```
User:   ZNDLU TYPE-6.2

System: NDLU0031I 17.25.47 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
                .APPA       6.2 A/P
                .APPC       6.2 A/P
                .AZAZ       6.2 I/P
                .AZAZZ001   6.2 I/P
                .AZAZZ002   6.2 I/P
                .AZAZZ003   6.2 I/P
                .AZAZZ004   6.2 I/P
                .AZAZZ005   6.2 I/P
                .AZAZZ006   6.2 I/P
                .AZAZZ007   6.2 I/P
                .AZAZZ008   6.2 I/P
                .AZAZZ009   6.2 I/P
                .AZAZZ010   6.2 I/P
                .AZAZZ011   6.2 I/P
        END OF ZNDLU DISPLAY
```

Information about a remote LU that has TPF/APPC sessions is displayed in the following example.

```
User:   ZNDLU ID-APPC

System: NDLU0032I 17.15.31 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D  TYPE STATUS  CLASS  CDRM
                .APPC        6.2 ACT/P           TPFB
        SCBID   MODENAME  P   PROC    APPL     LINK
        800001  SNASVCMG  L           APPA     TBCTCL1
        800002  TPFMOD1   W           APPA     TBCTCL1
        END OF ZNDLU DISPLAY
```

Information about sessions that were started for the LU named APPC and have a mode name of TPFMOD1 is displayed in the following example.

```
User:   ZNDLU ID-APPC MODE-TPFMOD1

System: NDLU0032I 17.15.43 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYPE STATUS  CLASS  CDRM
                .APPC       6.2 ACT/P           TPFB
        SCBID   MODENAME  P   PROC    APPL     LINK
        800002  TPFMOD1   W           APPA     TBCTCL1
        END OF ZNDLU DISPLAY
```

Information about a dynamic LU is displayed in the following example.

```
User:   ZNDLU ID-CNM02F98

System: NDLU0031I 15.27.08 BEGIN ZNDLU DISPLAY
        NETID   .LUNAME   D TYP STA CLASS PR APPL     CDRM     LINK
                .CNM02F98 Y  1  A   3270P    SMPD     VTAM2NCP N46E730
        NDLU0013I 15.27.08   LU - CNM02F98
        PCID - VTAMNET .VTAM2 . E383BE95 79744AE1
        END OF ZNDLU DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNDYN ADD–Create a SNA Resource Definition

Use this command to define the following network resources to the TPF system:
- Adjacent link station (ALS) resources
- Cross-domain resource manager (CDRM) resources
- Channel-to-channel (CTC) resources
- Network control program (NCP) resources.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- When defining a CTC resource, you must first define a CDRM resource for that CTC resource.

Format

```
►►──ZNDYN Add── ──┬─ALS-name──────────────────────────┬──────────────►◄
                  ├─CDRM-name── ──SUBarea-s────────────┤
                  ├─CTC-name── ──SUBarea-s─┬───────────┬┤
                  │                        └┤ Options ├┘│
                  └─NCP-name── ──SUBarea-s─┬───────────┬┘
                                           └┤ Options ├┘
```

Options:

```
   ┌ ──VRILto-0─┐  ┌ ──VRTO-0─┐  ┌ ──WINsize-100─┐
├──┴─────────────┴──┴───────────┴──┴───────────────┴──────────────┤
   └ ──VRILto-x─┘  └ ──VRTO-y─┘  └ ──WINsize-z───┘
```

ALS
: defines an ALS resource to the TPF system.

CDRM
: defines a CDRM resource to the TPF system.

CTC
: defines a CTC resource to the TPF system.

NCP
: defines an NCP resource to the TPF system.

name
: is the 1- to 8-character name of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

SUBarea-*s*
: is the subarea number for the resource, from 1–255.

Note: If you are creating a network definition for a CTC resource, specify the subarea number of the CDRM resource that was previously created for that CTC resource.

VRILto-*x*
is the number of SNA polling intervals for the virtual route input list timeout value, from 0–255.

Note: The SNA polling interval is defined by the SNAPOLL parameter in the SNAKEY macro.

VRTO-*y*
is the number of SNA polling intervals for the virtual route timeout value, from 0–65 535.

Note: The SNA polling interval is defined by the SNAPOLL parameter in the SNAKEY macro.

WINsize-*z*
is the number of path information units (PIUs), from 1–255, that the TPF system can send across the link before requiring a virtual route pacing response. This is referred to as the virtual route pacing window.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNDYN HELP
  ZNDYN ?
- Resource definitions are created only for the processor from which you enter this command. They are not created for each processor in a loosely coupled TPF system.
- If you perform an SNA fresh load, the resource definitions that you created using this command are deleted from the TPF system. Use the offline ACF/SNA table generation (OSTG) program to permanently define network resources to the TPF system.
- The NUMALS parameter in keypoint 2 (CTK2) specifies the maximum number of non-LU resource definitions that can exist in the TPF system. You can change this number using the SNAKEY macro.
- Resource definitions for remote LU resources are created automatically by the TPF system when those resources log on to the TPF system. You can also use the OSTG program to create resource definitions for remote LU resources.
- You can specify the parameters for this command in any order.

Examples

An ALS resource is defined to the TPF system in the following example.

```
User:   ZNDYN ADD ALS-ALSTEST

System: NDYN0080I 13.29.41 ALSTEST RESOURCE ADDED SUCCESSFULLY
```

An NCP resource is defined to the TPF system in the following example. The size of the virtual route pacing window is set to 250 PIUs. The default values for the VRILTO and VRTO parameters are used.

```
User:   ZNDYN ADD NCP-NCP1 SUB-52 WIN-250

System: NDYN0080I 13.30.53 NCP1 RESOURCE ADDED SUCCESSFULLY
```

Related Information

- See *TPF ACF/SNA Data Communications Reference* for more information about defining resources to the TPF system.
- See *TPF ACF/SNA Network Generation* for more information about the following:
  - SNA resource names
  - OSTG program
  - Virtual route input list timeout value (VRILTO)
  - Virtual route timeout value (VRTO)
  - SNA polling interval (SNAPOLL)
  - Virtual route pacing window (WINSIZE)
  - SNAKEY macro.

ZNDYN CHANGE–Change a Non-LU Resource

Use this command to change the name of a non-LU resource that is currently defined to the TPF system. You can change the name of the following types of non-LU resources:

- Adjacent link station (ALS) resources
- Cross-domain resource manager (CDRM) resources
- Channel-to-channel (CTC) resources
- Network control program (NCP) resources.

You can also change the subarea of a CDRM, CTC, or NCP resource.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You cannot change an active non-LU resource.
- You cannot change the name of the local system services control point (SSCP).
- You cannot change the subarea of a CDRM resource if CTC links are defined for that subarea.
- If you are changing the subarea of a CTC resource, a resource definition must exist for a CDRM resource with the new subarea.

Format

```
►►──ZNDYN Change── ──OLD-name── ──NEW-name── ─┬────────────┬──────────────►◄
                                              └─SUBarea-s──┘
```

OLD-*name*
: is the old name of the resource, where *name* is a 1- to 8-character resource name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

NEW-*name*
: is the new name of the resource, where *name* is a 1- to 8-character resource name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Notes:

  1. The new resource name cannot be the same as the old resource name.
  2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

SUBarea-*s*
is the new subarea number for the resource, from 1–255. If you do not specify a value for this parameter, the subarea is not changed.

Note: You can specify this parameter only when you change the name of a CDRM, CTC, or NCP resource.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNDYN HELP
  ZNDYN ?
- The resource definitions are changed only on the processor from which you enter this command. They are not changed on each processor in a loosely coupled TPF system.
- You can specify the parameters for this command in any order.

Examples

The name of the ALSTEST resource is changed to ALS1 in the following example.

```
User:   ZNDYN CHANGE OLD-ALSTEST NEW-ALS1

System: NDYN0074I 14.09.57 RESOURCE NAME ALSTEST WAS CHANGED TO ALS1
```

The name of the NCP7 resource is changed to NCPNEW in the following example. The subarea is also changed from 77 to 56.

```
User:   ZNDYN CHANGE OLD-NCP7 NEW-NCPNEW SUB-56

System: NDYN0074I 13.31.13 RESOURCE NAME NCP7 WAS CHANGED TO NCPNEW
        NDYN0075I 13.31.13 SUBAREA FOR NCPNEW WAS CHANGED FROM 77 TO 56
```

Related Information

- See *TPF ACF/SNA Network Generation* for more information SNA resource names.
- See *TPF ACF/SNA Data Communications Reference* for more information about SNA resources and subareas.

ZNDYN DISPLAY–Display Resource Name Hash (RNH) Tables

Use this command to display the following information from the resource name hash (RNH) tables:

- Statistical information about the distribution of resources in the RNH tables
- Information about a specific network resource
- Information about the resources on the resource name hash entry table (RNHET) synonym chain for a specific resource.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNDYN Display── ──┬─ALL──────────────────────────┬──────────────────────────►◄
                      ├─NAME── ──┬────────┬─name─────┤
                      │          └─netid.─┘          │
                      ├─RID── ──x────────────────────┤
                      └─SYNonym── ──┬────────┬─name──┘
                                    └─netid.─┘
```

ALL
displays statistical information about the distribution of the resources in the RNH tables.

NAME *netid***.***name*
displays information from the RNH tables about the specified resource, where *netid*.*name* is the 1- to 17-character network qualified name of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you do not specify a network identifier (*netid*) for the resource name, information about each resource with the specified name is displayed.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

RID *x*
displays information from the RNH tables about the resource with the specified resource identifier (RID) where *x* is the RID.

Note: You must specify a RID. You cannot specify a session control block identifier (SCBID).

SYNonym *netid***.***name*
displays information from the RNH tables about the resources on the RNHET synonym chain for the specified resource, where *netid*.*name* is the 1- to 17-character network qualified name of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If the specified resource does not exist, the RNHET synonym chain where the resource would be added, if it was defined, is displayed.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNDYN HELP
ZNDYN ?

Examples

Statistical information about the distribution of resources in the RNH tables is displayed in the following example.

```
User:   ZNDYN DISP ALL

System: NDYN0020I 10.17.02 *** ZNDYN DISPLAY ALL ***

          HETS DEFINED-     2998  IN USE-     2268  PERCENT INUSE -  76
          HPTS DEFINED-      293
          LARGEST NUMBER OF HETS CHAINED TO AN HPT -       16
          AVERAGE NUMBER OF HETS CHAINED TO AN HPT -        8
          AVAILABLE LIST
             TOP- 015861D0   BOTTOM- 01583CF0   COUNT-      474
          TERMINATION LIST
             TOP- 01583D18   BOTTOM- 015853C0   COUNT-      146

          *** END OF DISPLAY ***
```

Information from the RNH tables about TPFB.DYNA is displayed in the following example.

```
User:   ZNDYN DISPLAY NAME TPFB.DYNA

System: NDYN0030I 15.39.17 *** ZNDYN DISPLAY NAME ***

          TPFB.DYNA
             HET ADDRESS - 013A6F48
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 013A5080
             TERMINATION LIST
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
             RVT INFORMATION
               RID- 000996    RV1- 01354DC0    RV2- 0137E2C0
             HPT ADDRESS - 013AD7F0
               FIRST HET -     01393038    COUNT -       13

          *** END OF DISPLAY ***
```

Information from the RNH tables about TESTLU is displayed in the following example. Notice that more than one resource has the name TESTLU.

```
User:   ZNDYN DISPLAY NAME TESTLU

System: NDYN0030I 10.17.04 *** ZNDYN DISPLAY NAME ***

          TEST1.TESTLU
             HET ADDRESS - 01580000
               FORWARD CHAIN - 015886B0    BACKWARD CHAIN - 0157E7F0
             TERMINATION LIST -
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
             RVT INFORMATION
               RID- 000801    RV1- 0145E0A0    RV2- 01494020
             HPT ADDRESS - 0158A820
               FIRST HET -     0156EAD0    COUNT -       8

          TEST2.TESTLU
             HET ADDRESS - 015886B0
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 01580000
             TERMINATION LIST -
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
             RVT INFORMATION
               RID- 000B5F    RV1- 0147FB60    RV2- 0149ABE0
             HPT ADDRESS - 0158A820
               FIRST HET -     0156EAD0    COUNT -       8

          *** END OF DISPLAY ***
```

Information from the RNH tables about the resource with RID 9FE is displayed in the following example. Notice that the resource is on the RVT termination list.

```
User:   ZNDYN DISPLAY RID 9FE

System: NDYN0040I 10.17.04 *** ZNDYN DISPLAY RID ***

          DLU02    HET ADDRESS - 01584F88
               FORWARD CHAIN - 01586428    BACKWARD CHAIN - 01581BD0
             TERMINATION LIST -
               FORWARD CHAIN - 01584F38    BACKWARD CHAIN - 01584F60
             HPT ADDRESS - 0158A778
               FIRST HET -     0156F890    COUNT -   15
             RID- 9FE    RV1- 01471EC0    RV2- 01497FC0

          *** END OF DISPLAY ***
```

Information from the RNH tables about the resources on the RNHET synonym chain for the TPFB resource is displayed in the following example.

```
User:   ZNDYN DISPLAY SYN TPFB

System: NDYN0050I 15.39.17 *** ZNDYN DISPLAY SYNONYM ***

          TPFB
             HET ADDRESS - 0138F028
               FORWARD CHAIN - 013933D0    BACKWARD CHAIN - 00000000
             TERMINATION LIST
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
             RVT INFORMATION
               RID- 000002    RV1- 012F5140    RV2- 0136B040
             HPT ADDRESS - 013AD320
               FIRST HET -     0138F028    COUNT -       3

          GAMB
             HET ADDRESS - 013933D0
               FORWARD CHAIN - 01395D38    BACKWARD CHAIN - 0138F028
             TERMINATION LIST
               FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
             RVT INFORMATION
               RID- 0001B3    RV1- 01305FE0    RV2- 0136E660
              HPT ADDRESS - 013AD320
                FIRST HET -     0138F028    COUNT -       3

           L621B001
              HET ADDRESS - 01395D38
                FORWARD CHAIN - 00000000    BACKWARD CHAIN - 013933D0
              TERMINATION LIST
                FORWARD CHAIN - 00000000    BACKWARD CHAIN - 00000000
              RVT INFORMATION
                RID- 0002BC    RV1- 01310580    RV2- 01370780
              HPT ADDRESS - 013AD320
                FIRST HET -     0138F028    COUNT -       3

  *** END OF DISPLAY ***
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the RNH tables.

ZNDYN RECYCLE–Recycle RVT Entries

Use this command to recycle the RVT entries that are currently on the RVT termination list and make them available for use by new logical units (LUs) that are logging on to the TPF system.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNDYN RECYCLE──────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNDYN HELP
  ZNDYN ?
- Enter **ZNDYN DISPLAY ALL** to determine the number of RVT entries that are currently on the RVT termination list.
- This command recycles the RVT entries regardless of the recycle time defined by the DYNTO parameter in the SNAKEY macro.
- When a session is ended for an LU resource, the RVT entry for that LU resource is placed on the RVT termination list. If a session is started again for the LU resource, its RVT entry is removed from the RVT termination list and reused. If you use this command to recycle the RVT entries on the RVT termination list, the RVT entries are returned to the TPF system and can be used by different LU resources.

Examples

The RVT entries are recycled in the following example.

```
User:   ZNDYN RECYCLE

System: NDYN0005I 10.17.02 RVTS RECYCLED FROM TERMINATION LIST -      150
                           RVTS NOW ON AVAILABLE LIST -              709
```

Related Information

- See *TPF ACF/SNA Data Communications Reference* for more information about the RVT termination list and recycling RVT entries.
- See *TPF ACF/SNA Network Generation* for more information about the SNAKEY macro.

ZNERR–Display or Zero 37x5 NSC Error Counter

Use this command to display the current value of the 37*x*5 native subchannel (NSC) error counters or to reset the 37*x*5 NSC error counters to 0.

Requirements and Restrictions

None.

Format

```
►►──ZNERR── ──┬─DSPY─┬── ──┬─ccu─┬──────────────────────────────────►◄
              └─ZERO─┘     └─ALL─┘
```

DSPY
displays the current value of one or all of the 37*x*5 NSC error counters.

ZERO
resets one or all of the 37*x*5 NSC error counters to 0.

ALL
displays or resets the error counters for all the 37*x*5 NSCs defined in the TPF system.

ccu
is a hexadecimal NSC control unit and device address; for example, 01E.

Additional Information

The NSC error counters are maintained in the 37*x*5 keypoint record.

Examples

The error counters for all of the 37*x*5 NSCs in the TPF system are displayed in the following example.

```
User:   ZNERR DSPY ALL

System: NERR00 08.34.40
         NSC  ERR.CNT
        -------------
         01C   00000
         01A   00000
```

The error counter for the 01C 37*x*5 NSC is reset to 0 in the following example.

```
User:   ZNERR ZERO 01C

System: NERR00 08.35.28
         CNTS HAVE BEEN ZEROED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNETW ACT–Activate a Network Addressable Unit (NAU)

Use this command to activate one or more network addressable units (NAUs).

Requirements and Restrictions

- To activate a remote Advanced Peer-to-Peer Networking (APPN) control point (CP), the TPF system must be in NORM state and CP-CP sessions cannot already exist with any processor in the TPF loosely coupled complex.
- To activate any other NAU, the TPF system must be in CRAS state or higher.

Format

```
>>--ZNETW ACT-- ------------------------------------------------------------->

>--Id--+-+---------+--nau--+---------------------------------------------+-+--><
       | '-netid.--'       +--Sda-addr-----------------------------------+ |
       |                   '--Logon-appl--+--------------+--+-----------+-' |
       |                                  '--Cdrm-cdrmname-' '--CNT-count-'  |
       +-CTCNODES-----------------------------------------------------------+
       +-ALSNODES-----------------------------------------------------------+
       +-+---------+--cpname--+--------------+------------------------------+
       | '-netid.--'          '--LINK-alsname-'                             |
       '-CPCP---------------------------------------------------------------'
```

Id specifies the NAU to be activated, which can be one of the following:

netid.nau
: activates the specified resource, where *netid.nau* is the 1- to 17-character symbolic name of the NAU to be activated. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CTCNODES
: activates all channel-to-channel (CTC) links.

ALSNODES
: activates all PU 2.1 links.

netid.cpname
: activates CP-CP sessions with a remote CP, where *netid.cpname* is a 1- to 17-character symbolic name of the remote CP to be activated. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. You cannot specify the name of a remote CP that resides in an APPN end node (EN).
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each

dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CPCP
activates CP-CP sessions with a remote CP that is selected by the TPF system.

Sda-*addr*
activates a local NCP or CTC, where *addr* is the symbolic device address of the 37*x*5 NCP or CTC.

Note: This parameter is required for NCP but optional for CTC. When activating a CTC link, the TPF system chooses an available CTC SDA if you omit this parameter.

Logon-*appl*
activates cross-domain applications to local TPF application sessions, where *appl* is the symbolic name of the local TPF application that is placed in session for the cross-domain application specified on the ID parameter.

Cdrm-*cdrmname*
starts FMMR-FMMR or application-application (APPL-APPL) sessions through a gateway SNA network interconnection (SNI), where *cdrmname* is the 1- to 8-character symbolic name of the gateway system services control point (SSCP). The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. For FMMR-FMMR sessions, the remote FMMR is placed in session with the local FMMR through the specified CDRM.
2. If you specify the CDRM parameter, you must also specify the LOGON parameter.
3. The ID parameter must specify the remote FMMR or the remote application; the LOGON parameter must specify the local SSCP or the local application.
4. If you do not specify the CDRM parameter, how the TPF system starts the session with the specified remote LU is determined by the following:
   - Whether the TPF system is connected to the network as a PU 5 node, PU 2.1 node, or both.
   - How sessions with the specified remote LU were activated previously.

   See *TPF ACF/SNA Data Communications Reference* for more information about how the TPF system selects the path to use when starting a new LU-LU session.
5. Sessions across CTC links do not cross gateway boundaries. Specify the name of the remote CDRM to start an application-application (APPL-APPL) session across a CTC link to another TPF system or VTAM system.
6. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CNT-*count*
specifies the number of secondary logical unit (SLU) threads to be activated, where *count* is a number between 1 and 255.

Note: If you do not specify the CNT parameter, all SLU threads that are not already in session will be activated.

LINK-*alsname*
specifies the 1- to 8-character name of the adjacent link station (ALS) through which the CP-CP sessions will be established. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you do not specify the LINK parameter, the TPF system will search all of the active APPN links that support CP-CP sessions to find a link connected to the specified remote CP, and use the first link found to start the CP-CP sessions.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNETW HELP
ZNETW ?

Examples

A CTC link is activated in the following example.

```
User:   ZNETW ACT ID-TBCTCL1

System: NETW0011I 08.37.29 CTC TBCTCL1 ACT PROCESSING BEGUN
        XID20000I 08.37.29 SDA 03AF   NODE NAME TBCTCL1
        SUCCESSFUL XID FORMAT 2 COMPLETED
        NETW0033I 08.37.29 CTC TBCTCL1 ACT PROCESSING COMPLETE
```

An LU-LU session is started in the following example.

```
User:   ZNETW ACT ID-RBS1B001,LOGON-RCS1

System: NETW0011I 08.43.43 LU RBS1B001 ACT PROCESSING BEGUN
        NETW0033I 08.43.43 LU RBS1B001 ACT PROCESSING COMPLETE
```

In the following example, an LU-LU session is started through the specified gateway SNI.

```
User:   ZNETW ACT ID-RBS1B002,LOGON-RCS1,CDRM-TPFB

System: NETW0011I 08.44.14 LU RBS1B002 ACT PROCESSING BEGUN
        NETW0033I 08.44.15 LU RBS1B002 ACT PROCESSING COMPLETE
```

APPN CP-CP sessions are started in the following example. The TPF system selects the remote CP to use.

```
User:   ZNETW ACT ID-CPCP

System: NETW0100I 17.19.04 CPCP ACT PROCESSING BEGUN
        NETW0101I 17.19.14 CPCP ACT PROCESSING COMPLETE, CP VTAMNET.VTAM2 SELECTED
```

A specified number of SLU threads are activated in the following example.

```
User:   ZNETW ACT ID-RBS1 LOGON-RAS1 CNT-2 CDRM-TPFB

System: NETW0011I 18.13.09 APPL RAS1 ACT PROCESSING BEGUN
        NETW0067I 18.13.09 2 SLU SESSIONS REQUESTED, 2 INITIATED
        NETW0033I 18.13.09 APPL RAS1 ACT PROCESSING COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNETW DISPLAY–Display Network Subarea Status

Use this command to display the status of the network subareas.

Requirements and Restrictions

None.

Format

```
►►──ZNETW Display──────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNETW HELP
  ZNETW ?
- Use the ZNDLU command to display the status of individual network addressable units (NAUs).

Examples

The status of the network subareas is displayed in the following example.

```
User:   ZNETW D

System: NETW0060I 08.44.42 SUBAREA DISPLAY
        NAU       TYPE      STATUS     IN-PROG
        VTAM2CTC  CDRM      INACTV     P-TRC
        VTAM3CTC  CDRM      INACTV     P-TRC
        VTAM1     CDRM      INACTV     P-TRC
        VTAM2     CDRM      INACTV     P-TRC
        VTAM6CTC  CDRM      INACTV     P-TRC
        VTAM3     CDRM      INACTV     P-TRC
        VTAM6     CDRM      INACTV     P-TRC
        TPFA      CDRM      INACTV     P-TRC
        TPFB      CDRM      ACTIVE     P-TRC
        TPFC      SSCP      ACTIVE     P-TRC
        TPFD      CDRM      INACTV     P-TRC
        TPFE      CDRM      INACTV     P-TRC
        TPFZ      CDRM      INACTV     P-TRC
        TPF0      CDRM      INACTV     P-TRC
        N30E610   NCP       INACTV     P-TRC
        N34E610   NCP       INACTV     P-TRC
        N42E431   NCP       INACTV     P-TRC
        N46E431   NCP       INACTV     P-TRC
        N50E610   NCP       INACTV     P-TRC
        N58E431   NCP       INACTV     P-TRC
        END OF SUBAREA DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNETW INACT–Deactivate a Network Addressable Unit (NAU)

Use this command to deactivate a network addressable unit (NAU) and all its subordinate NAUs.

Requirements and Restrictions

The TPF system must be in CRAS state or higher.

Format

```
►►──ZNETW INACT── ──────────────────────────────────────────────────────►

►─Id-──┬──────────┬─nau─┬──────────────────────────────┬─┬──────────────►◄
       │ └─netid.─┘     └─┬─,Immediate─┬─┬───────────┬─┘ │
       │                  └─,Force─────┘ └─,SAVEsess─┘   │
       ├─CTCNODES────────────────────────────────────────┤
       ├─ALSNODES────────────────────────────────────────┤
       └─CPCP─┬─────────┬────────────────────────────────┘
              └─,Force──┘
```

Id specifies the NAU to be deactivated, which can be one of the following:

netid.nau
deactivates the specified NAU, where *netid.nau* is 1- to 17-character symbolic NAU name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CTCNODES
deactivates all active channel-to-channel (CTC) links.

ALSNODES
deactivates all PU 2.1 links.

Note: If you specify the ALSNODES option for the ID parameter, you cannot activate any PU 2.1 links on this processor until you enter **ZNETW ACT ID-ALSNODES** to allow PU 2.1 link activation again.

CPCP
deactivates the APPN CP-CP sessions.

Note: You must specify the FORCE parameter if you enter this command from a TPF processor in the loosely coupled complex that does not own the CP-CP sessions.

Immediate
immediately deactivates the specified NAU.

Force
forces the deactivation of the specified NAU.

SAVEsess
performs a nondisruptive cross-domain resource manager (CDRM) deactivation; that is, no LU-LU sessions are ended.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNETW HELP
  ZNETW ?
- If you do not specify the IMMEDIATE or FORCE parameter, a normal shutdown is performed.
- If you deactivate a local or remote LU, all sessions with that LU are ended.
- If you deactivate a local primary LU (PLU) that has secondary LU (SLU) threads, all sessions with the SLU threads (except TPF/APPC SLU thread sessions) are ended.
- If the local PLU is a TPF/APPC LU with SLU threads, you must deactivate each SLU thread separately.
- When you perform a nondisruptive deactivation on a CDRM session where the system services control point (SSCP) supports DACTCDRM type 4, all associated LU-LU sessions are disassociated from their controlling CDRM session. Both SSCPs do not remember the earlier relationship between the LU-LU sessions and the nondisruptively deactivated CDRM session when the CDRM session is restarted.
- If the TPF processor that owns the CP-CP sessions is deactivated, you can enter **ZNETW INACT ID-CPCP,FORCE** from another TPF processor in the loosely coupled complex. This command clears the processor shared value that indicates which processor owns the CP-CP sessions; however, it does ***not*** end the CP-CP sessions. Therefore, if the network still thinks the CP-CP sessions are active, you must deactivate the CP-CP sessions from the operator console of the remote CP. After the CP-CP sessions with the deactivated TPF processor have ended, new CP-CP sessions can be established with a different TPF processor in the loosely coupled complex.

Examples

A NAU is deactivated in the following example.

```
User:   ZNETW INACT ID-RCS1

System: NETW0011I 08.47.21 APPL RCS1 INACT PROCESSING BEGUN
        NETW0033I 08.47.28 APPL RCS1 INACT PROCESSING COMPLETE
```

An NAU is forcibly deactivated in the following example.

```
User:   ZNETW INACT ID-RBS1B001,F

System: NETW0011I 08.46.56 LU RBS1B001 INACT,F PROCESSING BEGUN
        NETW0033I 08.46.56 LU RBS1B001 INACT,F PROCESSING COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNETW MOUNT–Mount a PU 2.1 Channel Adapter

Use this command to mount a PU 2.1 channel adapter for an adjacent link station (ALS). This command allows you to dynamically mount symbolic device addresses (SDAs) that were added to the TPF configuration after the last initial program load (IPL) was performed.

Requirements and Restrictions

None.

Format

```
►►──ZNETW Mount── ──Sda-addr──┬────────┬──────────────────────────────►◄
                              └─,Force─┘
```

Sda-*addr*
: is the symbolic device address of a 37x5 device.

Force
: mounts the specified SDA by force. If the SDA is already mounted, it is dismounted and then mounted again.

Note: Dismounting the SDA of a link station disrupts communications on that link station.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNETW HELP
ZNETW ?

Examples

In the following example, the specified SDA is mounted.

```
User:   ZNETW MOUNT SDA-864

System: NETW0013I 17.10.12 SDA 0864 MOUNT PROCESSING BEGUN
        NETW0003I 17.10.12 SDA 0864 MOUNT PROCESSING COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNETW ROUTE–Display Route Information

Use this command to display explicit routing, virtual routing, and virtual route pacing information for a network control program (NCP) or channel-to-channel (CTC) resource.

Requirements and Restrictions

None.

Format

```
►►──ZNETW Route── ──Id-nau──────────────────────────────────────────►◄
```

Id-*nau*
is the 1- to 8-character symbolic network addressable unit (NAU) name of an NCP or CTC link. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNETW HELP
ZNETW ?

Examples

Routing and pacing information is displayed for the TBCTCL1 link in the following example.

```
User:   ZNETW ROUTE ID-TBCTCL1

System: NETW0050I 08.50.23  ROUTE INFORMATION STATUS
          EXPLICIT ROUTE ACTIVE
          VIRTUAL ROUTE ACTIVE
          VIRTUAL ROUTE PACING RESPONSE RECEIVED
          VIRTUAL ROUTE SEQUENCE NUMBER - 001E
          CONFIRMED VIRTUAL ROUTE SEQ NUM - 001D
          VIRTUAL ROUTE RESYNC INOPERATIVE
        END OF ROUTE INFORMATION
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNKEY–Display/Alter the SNA Communications Keypoint

Use this command to display or change the value of the fields in the SNA communications keypoint (CTK2).

Note: Not all of the fields in CTK2 can be changed using the ZNKEY command. See Table 12 for more information.

Requirements and Restrictions

None.

Format

```
►►──ZNKEY── ──┬─ALL──────────────────┬──────────────────────────────►◄
              └─keyword──┬────────┬──┘
                         └─-value─┘
```

ALL
displays the value of all the fields in the CTK2 keypoint.

keyword
is the name of the field in CTK2 that you want to display or change. See Table 12 for a list of valid fields and descriptions.

value
is the new value you want to assign to the specified field. If you do not specify a value for this parameter, the current value is displayed.

The following table describes the fields in the CTK2 keypoint that you can display or change.

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
CLAWADP	Display the current value for the number of Common Link Access to Workstation (CLAW) adapters that you can define for TCP/IP offload support.			
CLAWFD	Display the current value for the number of file descriptors that you can define for TCP/IP offload support.			
CLAWIP	Display the current value for the number of local Internet Protocol (IP) addresses that you can define for TCP/IP offload support.			
CPNAME	Display the control point name (CPNAME) of the TPF system. This field is only defined when the TPF system is running in LEN mode. This field is not defined when the TPF system is running in APPN mode.			

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
CTCANS	Display or change the number of seconds that the TPF system waits before declaring the other side inoperative. This is equivalent to the NCP Automatic Network Shutdown (ANS) or VTAM Missing Interrupt Handler (MIH) functions.	2	32 767	Immediately
CTCRBFR	Display the number of 4-KB input buffers that the TPF system allocated for receiving data from CTC link stations. **Note:** This indicates the number of input buffers per CTC link.			
CTCWBFRS	Display the number of 4-KB output buffers that the TPF system allocated for sending data to CTC link stations. **Note:** This indicates the number of total output buffers for all CTC links.			
CTCTGANY	Always displays YES, which indicates that the TPF system supports any transmission group (TG=ANY) on CTC links.			
DYNTO	Display or change the recycle time for an RVT entry. The *recycle time* is the minimum amount of time that an RVT entry must remain on the RVT termination list before it can be reused by a new LU.	0	65 535	Immediately
FMHDR	Display the FM header data that prefixes the system-generated messages to 3600 LUs.			
HARDREC	Display or change whether the TPF system should resynchronize the virtual route (VR) sequence number for each Network Control Program (NCP) that uses FID4 following a software or hardware initial program load (IPL).	Yes or No		Immediately
HPFMMR	Display whether or not the base system is using the high performance functional management message router (HPFMMR) transmit process.			
HPRALIVE	Display or change the value of the high-performance routing (HPR) alive timer. See *TPF ACF/SNA Data Communications Reference* for more information about the alive timer.	1	65 535	Next restart
HPRMTSIZ	Display the number of 4-KB frames in the high-performance routing message table (HPRMT). See *TPF ACF/SNA Data Communications Reference* for more information about the HPRMT.			
HPRPST	Display or change the value of the high-performance routing (HPR) path switch timer. See *TPF ACF/SNA Data Communications Reference* for more information about the path switch timer. **Note:** Once the value of the path switch timer is set for a rapid transport protocol (RTP) connection, it is used for the life of that RTP connection. Therefore, when you specify a new value for this parameter, that new value is used only for new RTP connections that are started. Existing RTP connections continue to use the original value of the path switch timer.	1	65 535	Immediately

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
ILWPC	Display or change the percentage of common blocks that must be available in the TPF system. If the percentage falls below this value, the TPF system will not perform certain functions that require common blocks, such as sending virtual route (VR) pacing responses, until the minimum percentage is available again. This helps the TPF system avoid running out of critical resources. This parameter is also used by adaptive rate-based (ARB) pacing to regulate the rate at which traffic is sent across rapid transport protocol (RTP) connections. See *TPF ACF/SNA Data Communications Reference* for more information about ARB pacing and RTP connections. If you specify a value of 0 for this parameter, the TPF system will not check the percentage of common blocks available and will use a default value to calculate ARB pacing.	0	99	Immediately
ILWPE	Display or change the percentage of entry control blocks (ECBs) that must be available in the TPF system. If the percentage falls below this value, the TPF system will not perform certain functions that require common blocks, such as sending virtual route (VR) pacing responses, until the minimum percentage is available again. This helps the TPF system avoid running out of critical resources. This parameter is also used by adaptive rate-based (ARB) pacing to regulate the rate at which traffic is sent across rapid transport protocol (RTP) connections. See *TPF ACF/SNA Data Communications Reference* for more information about ARB pacing and RTP connections. If you specify a value of 0 for this parameter, the TPF system will not check the percentage of ECBs available and will use a default value to calculate ARB pacing.	0	99	Immediately

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
ILWPF	Display or change the percentage of frame blocks that must be available in the TPF system. If the percentage falls below this value, the TPF system will not perform certain functions that require common blocks, such as sending virtual route (VR) pacing responses, until the minimum percentage is available again. This helps the TPF system avoid running out of critical resources. This parameter is also used by adaptive rate-based (ARB) pacing to regulate the rate at which traffic is sent across rapid transport protocol (RTP) connections. See *TPF ACF/SNA Data Communications Reference* for more information about ARB pacing and RTP connections. If you specify a value of 0 for this parameter, the TPF system will not check the percentage of frame blocks available and will use a default value to calculate ARB pacing.	0	99	Immediately
ILWPI	Display or change the percentage of input/output blocks (IOBs) that must be available in the TPF system. If the percentage falls below this value, the TPF system will not perform certain functions that require common blocks, such as sending virtual route (VR) pacing responses, until the minimum percentage is available again. This helps the TPF system avoid running out of critical resources. This parameter is also used by adaptive rate-based (ARB) pacing to regulate the rate at which traffic is sent across rapid transport protocol (RTP) connections. See *TPF ACF/SNA Data Communications Reference* for more information about ARB pacing and RTP connections. If you specify a value of 0 for this parameter, the TPF system will not check the percentage of IOBs available and will use a default value to calculate ARB pacing.	0	99	Immediately

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
ILWPS	Display or change the percentage of system work blocks (SWBs) that must be available in the TPF system. If the percentage falls below this value, the TPF system will not perform certain functions that require common blocks, such as sending virtual route (VR) pacing responses, until the minimum percentage is available again. This helps the TPF system avoid running out of critical resources. This parameter is also used by adaptive rate-based (ARB) pacing to regulate the rate at which traffic is sent across rapid transport protocol (RTP) connections. See *TPF ACF/SNA Data Communications Reference* for more information about ARB pacing and RTP connections. If you specify a value of 0 for this parameter, the TPF system will not check the percentage of SWBs available and will use a default value to calculate ARB pacing.	0	99	Immediately
INREC	Display or change whether or not a safe store file copy of recoverable input messages should be created.	Yes or No		Immediately
IPMTSIZE	Display the size of the Internet Protocol (IP) message table (IPMT) in 4-KB blocks.			
IPRBUFFS	Display the number of read buffers assigned to each read channel program for an IP router using channel data link control (CDLC) support.			
IPRBUFSZ	Display the size of each read buffer for IP routers using channel data link control (CDLC) support.			
IPTOS	Display or change the type of service (TOS) value to use for the network priority of outbound TPF IP packets.	0	255	Immediately
IPTRCNUM	Display the maximum number of individual IP traces that can be defined.			
IPTRCSIZ	Display the size of each individual IP trace table, defined in 4-KB blocks.			
LENNETID	Display the network ID used by the TPF system when it is attached as a PU 2.1 node.			
LMSCTI	Display or change the time interval, in seconds, that is used to notify the logon manager (LM) of the number of sessions active through an ALS.	1	32 767	Next restart
LUBLKT	Display or change the time interval before the LU blocking package transmits queued output data. The time unit for the value that you specify is double the value of SNAPOLL.	1	100	Immediately
MAXALS	Display the maximum number of channel-attached 37*x*5 and 3174 devices that can be physically attached to the TPF system and running at the same time.			

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
MAXASCU	Display or change the maximum number of Mapping of Airline Traffic over Internet Protocol (MATIP) agent set control units (ASCUs) that may be connected to the TPF system at one time.	0	16 777 215	Next restart
MAXBFRU	Display the number of input buffers allocated by the TPF system to receive data from channel-attached 37*x*5 devices.			
MAXCCB	Display the number of conversation control block (CCB) entries in the TPF system.			
MAXCDRSC	Display the maximum number of cross-domain resources by processor.			
MAXCTC	Display the maximum number of channel-to-channel (CTC) links that can be active simultaneously.			
MAXHCT	Display or change the maximum number of hotcon table (HCT) entries that the TPF system can maintain between relational databases and the TPFAR feature. This value must be MAXCCB-2 + MAXSOCK.	0	65 535	Next restart
MAXHPRSA	Display the number of entries in the high-performance routing session address table (HPRSAT). This is the maximum number of high-performance routing (HPR) LU-LU sessions per TPF processor. See *TPF ACF/SNA Data Communications Reference* for more information about the HPRSAT.			
MAXIPCCW	Display the maximum number of IP routers that can be active on this TPF host.			
MAXMATIP	Display or change the maximum number of MATIP sessions that can exist at one time.	0	16 777 215	Next restart
MAXOSA	Display the maximum number of Open Systems Adapter (OSA)-Express connections that can be active on the TPF system.			
MAXPCID	Display the number of procedure correlation ID (PCID) table entries in the TPF system.			
MAXPRIM	Display the number of resource name hash prime table (RNHPT) entries defined in the TPF system.			
MAXRTE	Display or change the maximum number of IP routing table entries.	0	2 048	Next restart
MAXRTPCB	Display the number of entries in the rapid transport protocol control block (RTPCB) table. This is the maximum number of RTP connections per TPF processor. See *TPF ACF/SNA Data Communications Reference* for more information about the RTPCB table.			
MAXRVT	Display the maximum number of SNA resources that can be defined for the TPF system.			
MAXSCB	Display the maximum number of session control blocks (SCBs) that are defined to the TPF system for TPF/APPC sessions.			

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
MAXSDD	Display or change the maximum number of structured query language (SQL) database management systems (DBMS) that are defined to the TPF system in the ZSQLD commands.	0	80	Next restart
MAXSID	Display the maximum number of LEN sessions that can flow through any channel-attached 37*x*5 or 3174 device.			
MAXSMTB	Display or change the number of 4 KB storage areas that are used per I-stream to hold the SQL trace information.	0	5	Next restart
MAXSNF	Display the number of slots in the sequence number field table.			
MAXSOCK	Display the maximum number of sockets that can be active on this TPF host using TCP/IP native stack support.			
MAXSRT	Display the number of system recovery table (SRT) entries generated in a processor.			
MAXTPI	Display the number of transaction program instance control block (TPICB) entries that can be active in a TPF system.			
MQITRC	Display the size of the Message Queue Interface (MQI) trace table. The size of the table is represented in 4-KB page units.			
NETID	Display the network ID used by the TPF system when it is attached as a PU 5 node.			
NBLKLU	Display the number of control blocks available for blocking output messages that are destined for General Access to X.25 Transport Extensions (GATE)/Fast Transaction Processing Interface (FTPI) resources.			
NUMALS	Display the number of non-LU entries that are allocated in the resource vector table (RVT).			
OLDAPPL	Display or change whether the TPF system should support old applications and SNA 3270 terminals, and bypass the restriction for multiple chained output message segments.	Yes or No		Immediately
OSABUFF	Display the number of 64-KB read buffers assigned to each OSA-Express connection.			
PARACOS	Display the class of service name that the TPF system uses while building a CDINIT response for a TPF/APPC parallel session.			
PIUTAPEQ	Display or change the tape queue threshold value for writing the PIU trace table to a real-time tape.	20	255	Immediately
RECIT	Display or change the activation frequency of the SNA input recovery timeout program. The default timeout value is *m* × *sec*, where *m* specifies the multiplier used to calculate the recovery timeout, and *sec* specifies the resolution factor, in seconds, to calculate the recovery timeout.	1,1	255,255	Next restart

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
RECOP	Display or change the number of seconds that the TPF system waits for a response to an output data message to a 3270 printer before taking recovery action.	1	32 767	Next restart
RECOT	Display or change the number of seconds that the TPF system waits for a response to an output data message before taking recovery action.	1	32 767	Next restart
RSPTO	Display or change the number of seconds that the TPF system waits for a response to a SNA command before the response is considered lost.	1	32 767	Next restart
RTPRSYNC	Display or change whether the TPF system will use the rapid transport protocol (RTP) connection resynchronization process to attempt to keep RTP connections active after a hard initial program load (IPL) is performed. See *TPF ACF/SNA Data Communications Reference* for more information about the RTP connection resynchronization process.	Yes or No		Immediately
RVTCTRL	Display or change the number of RVT entries that are processed before giving up control of the CPU during the process of keypointing, validating, and timing out the entries.	0	8 388 607	Immediately
SINGMODE	Display the mode name used for TPF/APPC single sessions initiated by the TPF system.			
SLOWTIME	Display or change the amount of time, in SNA polling intervals, that an adjacent link station (ALS) is allowed to be in slowdown before the TPF system deactivates the link. The SNAPOLL parameter defines the SNA polling interval. The SLOWTIME parameter also defines the amount of time, in 10-ms intervals, that an Internet Protocol (IP) router is allowed to be in slowdown mode before the TPF system deactivates the link.	0	32 767	Immediately
SNAPOLL	Display or change the SNA polling interval. The value can range from 1 to 5, where 1 represents 10 milliseconds (ms), 2 represents 20 ms, and so on, up to a maximum of 50 ms.	1	5	Immediately
SNDWN	Display or change the SNA shutdown level. If the number of available system work blocks (SWBs) in the TPF system falls below the number specified for this parameter, RVT scan will not issue any command. This value must be less than the value defined for the SNRST parameter.	11	32 766	Immediately
SNKEY	Display or change the time interval, in seconds, at which the SNA keypointable control blocks, for example, resource vector tables (RVTs), are written to file.	1	32 767	Next restart
SNQDPT	Display or change the SNA output queue depth.	8	50	Immediately

Table 12. Description of the Fields in the SNA Communications Keypoint (CTK2) (continued)

Field	Description	Minimum Value	Maximum Value	Change Takes Effect
SNRST	Display or change the SNA restart level. After the shutdown level is reached, this number specifies the number of SWBs that must be available in the TPF system before RVT scan can issue any commands. This value must be greater than the value defined for the SNDWN parameter.	12	32 767	Immediately
SNSESZ	Display the size of the in-core sense table.			
SNSETO	Display or change the time, in seconds, that the TPF system collects negative responses before issuing a sense message that indicates the total number of negative responses received during this time period.	0	360	Next restart
SOCKSWP	Display or change the interval, in minutes, in which the TCP/IP socket sweeper or the Secure Sockets Layer (SSL) sweeper will issue a CRETC macro to itself to close any inactive sockets. If you specify 0, the TCP/IP socket sweeper and the SSL sweeper are activated, but they do not close any inactive sockets or inactive shared SSL sessions. If you specify a value less than 3 but greater than 0, the CRETC interval is 3.	0	60	Immediately
SSLPROC	Display the number of SSL daemon processes.	0	16	
SSLTHRD	Display or change the number of thread ECBs per SSL daemon process.	0	32	Next restart of the SSL daemon
TPALLOC	Display or change the number of seconds that the TPF system waits for a remote LU to respond to a TPF/APPC ALLOCATE verb before timing out.	1	32 767	Immediately
TPRECV	Display or change the number of seconds that the TPF system waits for data after a TPF/APPC RECEIVE verb, or any verb that implies CONFIRM, before timing out.	1	32 767	Immediately
TPWAIT	Display or change the number of seconds that the TPF system waits for the remote LU to send information to any of the specified TPF/APPC conversations before timing out the TPPCC wait verb.	1	32 767	Immediately
TRANA	Display or change whether the user-written transaction analysis exit must be called for each input message.	Yes or No		Immediately
TRACSZ	Display the size of the PIU trace table. The size of the table is represented in 4 KB page units.			
UNITSZ	Display the size of the TPF input buffer used for transferring data from a 37*x*5 device.			
VRRTO	Display or change the timeout value, in seconds, for virtual route (VR) resynchronization.	0	120	Immediately

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNKEY HELP
ZNKEY ?

Examples

The value of the MAXALS field is displayed in the following example.

```
User:   ZNKEY MAXALS

System: NKEY0023I 08.51.28 CPUID C
        MAXALS   - 0000010
```

The value of the SNKEY field is changed in the following example.

```
User:   ZNKEY SNKEY-250

System: NKEY0025I 08.52.05 CPUID C
        SNKEY    - 250           WAS 0000060
```

Related Information

- See *TPF ACF/SNA Network Generation* for more information about the CTK2 keypoint and the SNAKEY macro.
- See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.
- See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP support.

ZNLM–Logon Manager Operator Command Entry

Use this command to send commands to the Logon Manager. The response is returned to the operator console and communications console when it is received from the Logon Manager.

Requirements and Restrictions

You can enter this command only in CRAS state or higher.

Format

```
►►──ZNLM── ─┬─────────┬─command──────────────────────────────────────►◄
            └─name,───┘
```

name
: is the 1- to 8-character name of an adjacent link station (ALS) or a control logical unit (CLU). The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. This name is used to find a CLU-CLU session over which to send the Logon Manager command. If you do not specify a name, the first TPF CLU found in the resource vector table (RVT) that is in session is used.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

command
: is the Logon Manager operator command. This command is passed unchanged and unedited to the Logon Manager.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNLM HELP
ZNLM ?

Examples

A command is sent to the Logon Manager in the following example.

```
User:   ZNLM CLUC02,INFO,ID=XXXX

System: NLM00001I 09.01.22 LM COMMAND ACCEPTED
        NLM00004I 09.01.22 COMMAND ISSUED TO LOGON MANAGER -
        INFO,ID=XXXX
        NLM00005I 09.01.22 RESPONSE FROM LOGON MANAGER -
        ELM010I INFO REQUEST ACCEPTED
        NLM00004I 09.01.22 COMMAND ISSUED TO LOGON MANAGER -
        INFO,ID=XXXX
        NLM00005I 09.01.22 RESPONSE FROM LOGON MANAGER -
        ELM014I NAME:    TYP:  STATUS:   CURRENT:     CONTROL:     REASON:  INITS:

        NLM00004I 09.01.22 COMMAND ISSUED TO LOGON MANAGER -
        INFO,ID=XXXX
        NLM00005I 09.01.22 RESPONSE FROM LOGON MANAGER -
        ELM040I XXXX     APPL  INACTIVE  LNKCNT=0     MINLNK=1             0
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the Logon Manager and SNA communications.

ZNMON–Display SNA Resource Statistics

Use this command to display statistical information about the active PU 2.1 LU-LU sessions, high-performance routing (HPR) rapid transport protocol (RTP) connections, and HPR LU-LU sessions through each adjacent link station (ALS).

You can also use this command to reset the maximum PU 2.1 LU-LU session count for each ALS to the number of PU 2.1 LU-LU sessions that are currently active.

The *maximum PU 2.1 LU-LU session count* indicates the maximum number of PU 2.1 LU-LU sessions that were active for an ALS at any given time. This number can be greater than the number of PU 2.1 LU-LU sessions that are currently active. For example, if four PU 2.1 LU-LU sessions were started and one was ended, the current number of active PU 2.1 LU-LU sessions is 3 and the maximum PU 2.1 LU-LU session count is 4.

The display also shows a list of active channel-to-channel (CTC) and network control program (NCP) resources.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNMON── ──ALS──┬────────┬──────────────────────────────────────►◄
                   └─,CLEAR─┘
```

ALS
displays statistical information about the active PU 2.1 LU-LU sessions, RTP connections, and HPR LU-LU sessions.

CLEAR
resets the maximum PU 2.1 LU-LU session count for each ALS to the number of PU 2.1 LU-LU sessions that are currently active.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZNMON HELP
ZNMON ?

Examples

The following information is displayed in the examples:

SDA
is the symbolic device address (SDA) of the ALS, NCP, or CTC resource.

NAME
is the node name of the ALS, NCP, or CTC resource.

SITSIZE
is the maximum number of PU 2.1 LU-LU sessions allowed for the ALS.

CURSESS
: is the current number of active PU 2.1 LU-LU sessions through the ALS.

MAXSESS
: is the maximum PU 2.1 LU-LU session count for the ALS.

MAXDATE
: is the date that the maximum PU 2.1 LU-LU session count occurred for the ALS.

MAXTIME
: is the time that the maximum PU 2.1 LU-LU session count occurred for the ALS.

RTPCONN
: is the current number of active RTP connections through the ALS.

HPRSESS
: is the current number of active HPR LU-LU sessions through the ALS.

Note: If the SNA resource is not an ALS, N/A is displayed for SITSIZE, CURSESS, MAXSESS, MAXDATE, MAXTIME, RTPCONN, and HPRSESS.

Statistical information about the active PU 2.1 LU-LU sessions, RTP connections, and HPR LU-LU sessions through each ALS, NCP, and CTC resource in the network is displayed in the following example.

```
User:   ZNMON ALS

System: NMON0002I 12.51.43 BEGIN ZNMON DISPLAY
        SDA     NAME   SITSIZE  CURSESS  MAXSESS  MAXDATE   MAXTIME  RTPCONN  HPRSESS
        0861  P30CA1B      800       13       34    21JUN  08.16.06        5       36
        0865  P34CA5B      800        1        2    21JUN  08.13.02        6       42
        03AF  TCCTCL1      N/A      N/A      N/A      N/A       N/A      N/A      N/A
        END OF ZNMON DISPLAY
```

The maximum number of PU 2.1 LU-LU sessions for each ALS is reset to the number of PU 2.1 LU-LU sessions that are currently active in the following example.

```
User:   ZNMON ALS,CLEAR

System: NMON0002I 12.51.43 BEGIN ZNMON DISPLAY
        SDA     NAME   SITSIZE  CURSESS  MAXSESS  MAXDATE   MAXTIME  RTPCONN  HPRSESS
        0861  P30CA1B      800       13       34    21JUN  08.16.06        5       36
        0865  P34CA5B      800        1        2    21JUN  08.13.02        6       42
        03AF  TCCTCL1      N/A      N/A      N/A      N/A       N/A      N/A      N/A
        MAXIMUM SESSION COUNT, DATE, AND TIME HAVE BEEN RESET
        END OF ZNMON DISPLAY

User:   ZNMON ALS

System: NMON0002I 12.51.43 BEGIN ZNMON DISPLAY
        SDA     NAME   SITSIZE  CURSESS  MAXSESS  MAXDATE   MAXTIME  RTPCONN  HPRSESS
        0861  P30CA1B      800       13       13    21JUN  08.19.42        5       36
        0865  P34CA5B      800        1        2    21JUN  08.19.42        6       42
        03AF  TCCTCL1      N/A      N/A      N/A      N/A       N/A      N/A      N/A
        END OF ZNMON DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about HPR support, RTP connections, and SNA communications.

ZNNCB–Initialize NCB Records

Use this command to initialize the node control block (NCB) records in the TPF system.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You cannot initialize the NCB records when the NCB reconciliation function or the NCB reorganization function is running.

  Note: You can enter the ZNNCB RECON command to end the NCB reconciliation function or the ZNNCB REORG command to end the NCB reorganization function.
- NCB records are initialized only if there are no active sessions for the logical unit (LU) resource.

Format

```
►►──ZNNCB── ──┬─ALL──────────────────────────────┬──────────────────────►◄
              ├─MOD── ──module───────────────────┤
              └─┬─NAME─┬─ ─┬────────┬──name──────┘
                └─NOD──┘   └─netid.─┘
```

ALL
: initializes all of the NCB records in the TPF system.

MOD *module*
: initializes all of the NCB records on a module, where *module* is the hexadecimal module number.

NAME or NOD
: initializes the NCB for a specific LU resource.

netid.*name*
: is the 1- to 17-character network qualified name of an LU resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

  Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNNCB HELP
  ZNNCB ?
- This command initializes both the fixed file and long-term pool file NCB records. LU resources created using the offline ACF/SNA table generation (OSTG) program use fixed file NCB records. All of the other LU resources use long-term pool file NCB records.

Examples

All of the NCB records in the TPF system are initialized in the following example.

```
User:   ZNNCB ALL

System: NNCB0016I 09.04.12 START INITIALIZING NCB
        NNCB0010I 09.05.16 INITIALIZATION COMPLETE - ACTIVE NODES SKIPPED
```

The NCB record for the specified LU resource is initialized in the following example.

```
User:   ZNNCB NOD PBPBB001

System: NNCB0001I 09.03.37 INITIALIZATION COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about NCB records.

ZNNCB DISPLAY–Display NCB Information

Use this command to display the following information about node control block (NCB) records:

- Information from the NCB control record, which includes:
  – Number of current NCB directory records defined
  – Record type for the current NCB directory records
  – Number of staged NCB directory records defined
  – Record type for the staged NCB directory records
  – Status information about the NCB reconciliation function
  – Status information about the NCB reorganization function.
- Information about the NCB records for a specific logical unit (LU) resource, which includes:
  – How the LU resource was created; either using the offline ACF/SNA table generation (OSTG) program or dynamic LU support.
  – For LU resources created using the OSTG program:
    - NCB ordinal number
    - NCB file address.
  – For LU resources created dynamically:
    - Ordinal number of the NCB directory record
    - Number of the NCB directory record entry
    - Address of each existing NCB record.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNNCB DISplay── ──┬─ALL──────────────────────────────┬──────────────────►◄
                      └─┬─NAME─┬─ ──┬─────────┬─name─┘
                        └─NOD──┘    └─netid.──┘
```

ALL
displays information from the NCB control record.

NAME or NOD
displays information about the specified LU resource.

*netid***.***name*
is the 1- to 17-character network qualified name of the LU resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNNCB HELP
ZNNCB ?

Examples

Information from the NCB control record is displayed in the following example. There are 293 current NCB directory records defined and they have a record type of NCBN4. There are 317 staged NCB directory records defined and they have a record type of NCBN5. The NCB reconciliation function is not active. The NCB reorganization function is active on CPU B.

```
User:   ZNNCB DISP ALL

System: NNCB0110I 11.57.26 *** NCB CONTROL RECORD DISPLAY ***

           CURRENT NCBDR
             NUMBER DEFINED      -        293
             RECORD TYPE         -      NCBN4

           STAGED NCBDR
             NUMBER DEFINED      -        317
             RECORD TYPE         -      NCBN5

           RECON STATUS          - NOT ACTIVE
             LAST COMPLETED      - 95/04/10 12.27.19

           REORG STATUS          - ACTIVE ON CPU B
             NCBDR COPY STATUS   - IN PROGRESS
             LAST STARTED        - 95/04/10 13.48.02
             LAST COMPLETED      -

           *** END OF DISPLAY ***
```

Information about the NCB record for the PBPB resource is displayed in the following example. PBPB was created using the OSTG program, which indicates that the NCB record is a fixed file record. The ordinal number for the NCB record is 282 and the file address for the NCB record is FC040A08.

```
User:   ZNNCB DISPLAY NAME PBPB

System: NNCB0130I 12.00.09 NCB INFORMATION FOR PBPB

           CREATED USING OSTG
           NCB ORDINAL NUMBER    - 00000282
           NCB FILE ADDRESS      - FC040A08
           *** END OF DISPLAY ***
```

Information about the NCB records for the TEST.DYNA resource is displayed in the following example. TEST.DYNA was created using dynamic LU support, which indicates that the NCB records are long-term pool file records.

The ordinal number of the NCB directory record is 0FE, and the NCB information for TEST.DYNA is contained in the first entry in that NCB directory record. Two NCB records exist for the TEST.DYNA resource and the addresses of these NCB records are saved in NCB slots 2 and 5.

```
User:   ZNNCB DISPLAY NOD TEST.DYNA

System: NNCB0131I 10.32.34 NCB INFORMATION FOR TEST.DYNA

           CREATED DYNAMICALLY
           NCBDR ORDINAL NUMBER  - 0FE
           NCBDR ENTRY NUMBER    -   1

               NCB        NCB FILE ADDRESS
                2             000805AA
                5             00080208
           *** END OF DISPLAY ***
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the NCB control record, NCB directory records, NCB slots, and NCB records.

ZNNCB RECON–Reconcile NCB Directory Records

Use this command to reconcile the node control block (NCB) directory records in the TPF system when they are becoming full. This function can delay the need for increasing the number of NCB directory records if the largest number of entries in an NCB directory record is reduced to an acceptable number.

Reconciling the NCB directory records does the following:

- Returns to the TPF system the unused NCB records. An NCB record is considered unused if the logical unit (LU) resource is no longer in session and no output message transmission (OMT) queue exists for that NCB record.
- Returns to the TPF system the unused entries in the NCB directory records. An NCB directory record entry is considered unused if no NCB records exist for that NCB directory record entry.

You can reconcile all of the NCB directory records or only the NCB directory record entry for a specific LU.

The following information is displayed when the NCB reconciliation function has been completed:

- Largest number of entries in an NCB directory record
- Average number of entries in an NCB directory record
- Number of long-term pool file NCB records returned to the TPF system
- Number of long-term pool file NCB records still in use.

Requirements and Restrictions

- You can enter this command only in NORM state.
- You cannot reconcile the NCB directory records when the NCB initialization function or the NCB reorganization function is running.

  Note: You can enter the ZNNCB REORG command to end the NCB reorganization function.

Format

```
►►──ZNNCB RECON── ──┬─ABORT─┬──────┬──────────────────┬───────────────────►◄
                    │       └─ ─BP─┘                  │
                    ├─ALL─────────────────────────────┤
                    └─┬─NAME─┬─ ─┬─────────┬─luname───┘
                      └─NOD──┘   └─netid.──┘
```

ABORT
cancels the NCB reconciliation function.

Note: You can specify this parameter only when you enter the ZNNCB RECON command from the processor where the NCB reconciliation function was started. Specify the BP parameter to cancel the NCB reconciliation function from a processor other than the one where it was started.

BP
allows you to cancel the NCB reconciliation function from a processor other than the one where it was started.

ALL
reconciles all of the NCB directory records in the TPF system.

NAME or NOD
reconciles the entry in the NCB directory record for the specified LU resource.

*netid***.***luname*
is the 1- to 17-character network qualified name of an LU resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNNCB HELP
  ZNNCB ?
- Each NCB directory record can contain as many as 84 entries.
- Enter **ZNNCB DISPLAY ALL** to display the current status of the NCB reconciliation function.
- If enough NCB directory record entries were not returned to the TPF system by the NCB reconciliation function, you can enter the ZNNCB REORG command to increase the number of NCB directory records.

Examples

All of the NCB directory records in the TPF system are reconciled in following example. Notice that after the NCB reconciliation function has been completed, the NCB directory records are still rather full. This indicates that you should increase the number of NCB directory records by using the NCB reorganization function.

```
User:   ZNNCB RECON ALL

System: NNCB0076I 16.16.16 NCB RECONCILIATION STARTED
        NNCB0077I 16.16.23 NCB RECONCILIATION COMPLETED
                               NUMBER OF NCBS KEPT                -    21519
                               NUMBER OF NCBS RETURNED            -      452
                               HIGHEST COUNT OF ENTRIES PER NCBDR -       79
                               AVERAGE COUNT OF ENTRIES PER NCBDR -       52
                           END OF DISPLAY
```

In the following example, the ZNNCB RECON command was entered to cancel the NCB reconciliation function from a processor other than the one where it was started. Notice that, in this example, you must specify the BP parameter.

```
User:   ZNNCB RECON ABORT

System: NNCB0073W 16.33.22 MUST SPECIFY BP PARAMETER TO ABORT RECON FROM THIS CPU

User:   ZNNCB RECON ABORT BP

System: NNCB0074I 16.33.45 NCB RECONCILIATION ABORT INITIATED
        NNCB0075I 16.33.45 NCB RECONCILIATION ABORTED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about NCB directory records and the NCB reconciliation function.

ZNNCB REORG–Reorganize NCB Directory Records

Use this command to change the number of NCB directory records in the TPF system. For example, you can increase the number of NCB directory records when the current NCB directory records are becoming full.

Reorganizing the NCB directory records is a 2-step process:

1. Enter **ZNNCB REORG START** to copy the current NCB directory records to the staged NCB directory records. This step redistributes the current NCB directory record entries among the different number of staged NCB directory records.
2. Enter **ZNNCB REORG SWITCH** from any processor in the loosely coupled TPF system to switch the current and the staged NCB directory records. This step makes the current NCB directory records the staged NCB directory records and the staged NCB directory records the current NCB directory records.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You cannot reorganize the NCB directory records when the NCB initialization function or the NCB reconciliation function is running.

  Note: You can enter the ZNNCB RECON command to end the NCB reconciliation function.
- You cannot reorganize the NCB directory records when the online file recoup function is running.

Format

```
►►──ZNNCB REORG── ──┬─ABORT──┬───────┬─┬──────────────────────────────────►◄
                    │        └─ ─BP──┘ │
                    ├─STart────────────┤
                    └─SWitch───────────┘
```

ABORT
cancels the NCB reorganization function.

Note: You can specify this parameter only when you enter the ZNNCB REORG command from the processor where the NCB reorganization function was started. Specify the BP parameter to cancel the NCB reorganization function from a processor other than the one where it was started.

BP
allows you to cancel the NCB reorganization function from a processor other than the one where it was started.

STart
starts the NCB reorganization function by copying the current NCB directory records to the staged NCB directory records.

SWitch
completes the NCB reorganization function by switching the current and staged NCB directory records.

Note: The TPF system prompts you to switch the NCB directory records when the copy process has been completed.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNNCB HELP
  ZNNCB ?
- Enter **ZNNCB DISPLAY ALL** to determine the number of current and staged NCB directory records that are defined. This command also indicates the record types for the current and staged NCB directory records, which alternate between #NCBN4 and #NCBN5, and displays status information about the NCB reorganization function.

Examples

The NCB directory records are reorganized in the following example. First, the START parameter is specified to copy the current NCB directory record entries to the staged NCB directory records. When the copy step was completed, the TPF system displayed a message to switch the NCB directory records by specifying the SWITCH parameter.

```
User:   ZNNCB REORG START

System: NNCB0070I 16.38.33 NCBN5   DATABASE INITIALIZATION STARTED
        NNCB0071I 16.38.40 NCBN5   DATABASE INITIALIZATION COMPLETED
        NNCB0093I 16.38.40 NCB REORG DATABASE COPY TASK BEGUN
        NNCB0097I 16.38.46 NCB REORG DATABASE COPY COMPLETED
                           ENTER ZNNCB REORG SWITCH

User:   ZNNCB REORG SWITCH

System: NNCB0085I 16.39.23 NCB REORG DATABASE SWITCH INITIATED
        NNCB0070I 16.39.23 NCBN4   DATABASE INITIALIZATION STARTED
        NNCB0071I 16.39.25 NCBN4   DATABASE INITIALIZATION COMPLETED
        NNCB0086I 16.39.25 NCB REORG DATABASE SWITCH COMPLETED
```

In the following example, the ZNNCB REORG command was entered to cancel the NCB reorganization function from a processor other than the one where the it was started. Notice that, in this example, you must specify the BP parameter.

```
User:   ZNNCB REORG ABORT

System: NNCB0083W 16.42.49 MUST SPECIFY BP PARAMETER TO ABORT REORG FROM THIS CPU

User:   ZNNCB REORG ABORT BP

System: NNCB0084I 16.42.56 ABORT INITIATED FOR NCB DATABASE REORG
        NNCB0070I 16.42.58 NCBN5   DATABASE INITIALIZATION STARTED
        NNCB0071I 16.43.03 NCBN5   DATABASE INITIALIZATION COMPLETED
        NNCB0098I 16.43.03 NCB REORG ABORTED
```

Related Information

- See *TPF ACF/SNA Data Communications Reference* for more information about NCB directory records, both current and staged, and the NCB reorganization function.
- See *TPF Database Reference* for more information about online file recoup.

ZNNCS–Display Network Command Status Table (NCST)

Use this command to display the network command status table (NCST). This table contains information about each network addressable unit (NAU) that is in the process of being activated or deactivated as a result of the ZNETW ACT or ZNETW INACT command.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNNCS──────────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example displays the NCST, where

NAUNAME
: is the name of the network addressable unit (NAU).

IN PROGRESS
: indicates the activity being performed on the NAU, where

  ACT
  : NAU is being activated.

  INACT
  : NAU is being deactivated normally.

  INACT I
  : NAU is being deactivated immediately.

  INACT F
  : NAU is being deactivated by force (aborted).

  SAVESESS
  : CDRM is being nondisruptively deactivated.

START AT
: is the time the activity being performed on the NAU was started.

```
User:   ZNNCS

System: NNCS0000I 09.17.46 BEGIN ZNNCS DISPLAY
        NAUNAME     IN PROGRESS       START AT
        FAST0A1     ACT               08.50.02
        LINE001D    INACT F           09.12.05
        SSCP2RMT    INACT I SAVESESS  09.15.20
        END OF ZNNCS DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNOPL BUILD–Rebuild RVT during Next IPL

Use this command to rebuild the resource vector table (RVT) by forcing an SNA fresh load from the current resource resolution table (RRT) during the next initial program load (IPL).

Requirements and Restrictions

None.

Format

```
►►──ZNOPL BUild──────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNOPL HELP
  ZNOPL ?
- The node control block (NCB) records are ***not*** initialized during the fresh load.
- A fresh load is performed only on the processor where you enter this command.

Examples

In the following example, a fresh load will be performed during the next IPL.

```
User:   ZNOPL BUILD

System: NOPL0069A 09.49.42 BUILD COMPLETE, ZRIPL TO FORCE FRESH LOAD FROM CURRENT RRT.

User:   ZRIPL
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the RVT, RRT, and fresh load function.

ZNOPL FALLBACK–Fallback to the Alternate RRT Definitions

Use this command to fall back from the current resource resolution table (RRT) definitions to the alternate RRT definitions. For example, you may want to fall back to the alternate RRT definitions if you perform a dynamic load (by specifying the DYNAMIC parameter for the ZNOPL LOAD command) and it does not contain the correct resource definitions.

Requirements and Restrictions

You can enter this command only when it is enabled by the TPF system. For example, you can enter this command after you perform a dynamic load (by specifying the DYNAMIC parameter for the ZNOPL LOAD command). To determine if the ZNOPL FALLBACK command is enabled, enter **ZNOPL STATUS**.

Format

```
►►──ZNOPL FAllback──┬─────────┬──────────────────────────────────────►◄
                    └──FRESH──┘
```

FRESH
forces a fresh load of the alternate RRT definitions. Specify this parameter if errors occurred when you entered **ZNOPL FALLBACK**.

Notes:

1. If you do not specify the FRESH parameter, a dynamic load of the alternate RRT definitions is performed.
2. The FRESH parameter ends all active sessions on the processors where you need to fall back to the alternate RRT definitions.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNOPL HELP
  ZNOPL ?
- When you fall back from the current RRT definitions to the alternate RRT definitions, the current RRT definitions are removed from the TPF system and the alternate RRT definitions become the current RRT definitions.
- Once the TPF system falls back to the alternate RRT definitions, enter the ZNOPL STATUS command to determine the processors where you must incorporate the alternate RRT definitions, which are now the current RRT definitions. You can incorporate the alternate RRT definitions on a processor by performing an initial program load (IPL) or entering the ZNOPL MERGE command on that processor.

  Note: If you specified the FRESH parameter, you must perform an IPL to incorporate the alternate RRT definitions. You cannot enter the ZNOPL MERGE command.

Examples

Processor B falls back to the alternate RRT definitions in the following example. The ZNOPL MERGE command was entered to incorporate the current RRT definitions, which used to be the alternate RRT definitions before the fallback function was performed, on processor B.

Notice that the ZNOPL status information indicates that the current RRT definitions must still be incorporated on processor C. Also notice that section 2 of the RRT is unused because the fallback function was performed.

```
User:   ZNOPL FALLBACK

System: NOPL0063A 12.03.03 FALLBACK COMPLETE ON CPU-B , DISPLAY STATUS AND ENTER ZNOPL
        MERGE IF NECESSARY.

User:   ZNOPL MERGE

System: NOPL0064I 12.06.17 ONLINE MERGE STARTED.
        NOPL0065I 12.06.17 ONLINE MERGE COMPLETE.

User:   ZNOPL STATUS

System: NOPL0002I 12.05.12 SNA RESOURCE DEFINITION STATUS
        RRT SECTION 1 IS CURRENT
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.47.41
        DESCRIPTION OSTGM2

        RRT SECTION 2 IS UNUSED
        UPDATE IS DISABLED
        FALLBACK IS DISABLED
        ONLINE MERGE IS DISABLED    IN THIS PROCESSOR
        BUILD IS NOT SCHEDULED FOR NEXT IPL
        CPUID   ACTIVE   USING CURRENT DEFINITION
          B       YES              YES
          C       YES              NO
          D       NO               NO
          E       NO               NO
          Z       NO               NO
          0       NO               NO
        DISPLAY COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about falling back to the alternate RRT definitions.

ZNOPL LOAD–Load SNA Network Definitions

Use this command to load offline SNA table generation (OSTG) network definitions from an input data set to the TPF system. You can perform a fresh load or a dynamic load.

A *fresh load* replaces the current resource definitions in the resource resolution table (RRT) with the new resource definitions from the OSTG input data set. When you perform an initial program load (IPL), the fresh load function rebuilds the resource vector table (RVT) using the new resource definitions. All active sessions are destroyed during a fresh load.

A *dynamic load* first loads the new resource definitions to the alternate RRT. When you enter the ZNOPL UPDATE command, the dynamic load function verifies that the new and current resource definitions are compatible by ensuring that no local or active resources are being deleted. When you enter the ZNOPL MERGE command, the dynamic load function merges the new resource definitions with the RVT; the RVT is ***not*** rebuilt. Therefore, an IPL is ***not*** required during a dynamic load and active sessions are maintained.

Requirements and Restrictions

- Verify that the input source is mounted with a data definition name of SNA. See “ZDSMG DEFINE–Define a Data Definition” on page 364 for information about defining data definition names. See “ZTMNT–Mount a Tape” on page 1353 for information about mounting a tape.
- Deactivate the SNA network before performing a fresh load.

Format

```
►►──ZNOPL── ──LOad── ──┬─DYnamic─┬── ──id──────────────────────────────►◄
                       └─FResh───┘
```

DYnamic
performs a dynamic load of the SNA resource definitions.

FResh
performs a fresh load of the SNA resource definitions.

id
is the pilot ID of the input data set for the OSTG definitions that you want to load.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNOPL HELP
  ZNOPL ?
- If you perform a fresh load, you must perform an IPL on each processor to incorporate the new resource definitions in the TPF system.
- If you perform a dynamic load, you must do the following after you enter the ZNOPL LOAD command:
  – Enter **ZNOPL UPDATE** to make the new RRT the current RRT.

– Enter **ZNOPL MERGE** or perform an IPL on each active processor to merge the new resource definitions in the RVT.

Note: If a processor is inactive when you perform a dynamic load, the RVT is automatically updated with the new resource definitions when an IPL is performed on that processor. There is no need to enter the ZNOPL MERGE command once the processor is activated.

- After you perform a dynamic load, you can fall back from the current RRT definitions to the alternate RRT definitions, if necessary, to by using the ZNOPL FALLBACK command.

  Note: You cannot fall back to the alternate RRT definitions after you perform a fresh load.

Examples

A dynamic load of SNA resources is performed from pilot tape X in the following example. Notice that you must enter the ZNOPL UPDATE command to continue the dynamic load function.

```
User:   ZTMNT SNA 424 AI BP

System: COTM0033W 12.20.42 TMNT HPN    TAPE SNA IS LABELED
                           VSN A00737 G0001 S0001 D38K   NL  NOBLK
        COTM0046I 12.20.42 TMNT HPN    TAPE SNA MOUNTED ON DEVICE 424
                           VSN A00737 G0001 S0001 D38K   NL  NOBLK

User:   ZNOPL LOAD DYNAMIC X

System: NOPL0061A 12.20.51 SUCCESSFUL DYNAMIC LOAD, UPDATE ENABLED.
        COTC0087A 12.20.51 TCLS HPN    REMOVE SNA FROM DEVICE 424
                           VSN A00737  NOBLK
```

A fresh load of SNA resources is performed from pilot tape B in the following example.

```
User:   ZTMNT SNA 424 AI BP

System: COTM0033W 12.20.42 TMNT HPN    TAPE SNA IS LABELED
                           VSN A00737 G0001 S0001 D38K   NL  NOBLK
        COTM0046I 12.20.42 TMNT HPN    TAPE SNA MOUNTED ON DEVICE 424
                           VSN A00737 G0001 S0001 D38K   NL  NOBLK

User:   ZNOPL LOAD FRESH B

System: NOPL0060A 16.06.58 FRESH LOAD COMPLETE ON CPU-B.  ZRIPL TO INCORPORATE
        NEW DEFINITIONS

User:   ZRIPL
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the dynamic and fresh load functions.

ZNOPL MERGE–Perform an Online Merge

Use this command to merge the new resource resolution table (RRT), which is now the current RRT, with the resource vector table (RVT) without requiring an initial program load (IPL) or disrupting the SNA network.

Requirements and Restrictions

- You can enter this command only after you enter the ZNOPL UPDATE command.
- You can enter this command only in 1052 state or higher.
- In a loosely coupled TPF system, you must enter this command on each active processor to incorporate the new RRT definitions in the RVT on that processor.

Format

```
►►──ZNOPL MErge──────────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNOPL HELP
  ZNOPL ?
- A message is displayed to describe any errors that occur during the online merge function. After you correct the errors, enter this command again to start the online merge function again.

Examples

The current RRT is merged with the RVT on processor B in the following example. Notice that the ZNOPL status information indicates that the current RRT still needs to be merged with the RVT on processor C.

```
User:   ZNOPL LOAD DYNAMIC B

System: NOPL0061A 12.20.51 SUCCESSFUL DYNAMIC LOAD, UPDATE ENABLED.
        COTC0087A 12.20.51 TCLS HPN    REMOVE SNA FROM DEVICE 424
                           VSN A00737  NOBLK
User:   ZNOPL UPDATE

System: NOPL0062A 12.27.39 UPDATE COMPLETE ON CPU-B , ENTER ZNOPL MERGE TO INCORPORATE
        NEW DEFINITIONS.

User:   ZNOPL MERGE

System: NOPL0064I 12.28.22 ONLINE MERGE STARTED.
        NOPL0065I 12.28.23 ONLINE MERGE COMPLETE.

User:   ZNOPL STATUS

System: NOPL0002I 12.51.57 SNA RESOURCE DEFINITION STATUS
        RRT SECTION 1 IS NEW/ALTERNATE
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.47.41
        DESCRIPTION OSTGM2

        RRT SECTION 2 IS CURRENT
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.26.23
        DESCRIPTION OSTGM1

        UPDATE IS DISABLED
        FALLBACK IS ENABLED
        ONLINE MERGE IS DISABLED    IN THIS PROCESSOR
        BUILD IS NOT SCHEDULED FOR NEXT IPL
        CPUID   ACTIVE   USING CURRENT DEFINITION
          B       YES             YES
          C       YES             NO
          D       NO              NO
          E       NO              NO
          Z       NO              NO
          0       NO              NO
        DISPLAY COMPLETE
```

The current RRT is merged with the RVT on processor C in the following example. Notice that a message is displayed on processor C to indicate that you must enter the ZNOPL MERGE command to incorporate the current RRT definitions.

```
System: NOPL0062A 12.54.00 UPDATE COMPLETE ON CPU-B , ENTER ZNOPL MERGE TO INCORPORATE
        NEW DEFINITIONS.

User:   ZNOPL MERGE

System: NOPL0064I 12.28.22 ONLINE MERGE STARTED.
        NOPL0065I 12.28.23 ONLINE MERGE COMPLETE.

User:   ZNOPL STATUS

System: NOPL0002I 12.51.57 SNA RESOURCE DEFINITION STATUS
        RRT SECTION 1 IS NEW/ALTERNATE
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.47.41
        DESCRIPTION OSTGM2

        RRT SECTION 2 IS CURRENT
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.26.23
        DESCRIPTION OSTGM1

        UPDATE IS DISABLED
        FALLBACK IS ENABLED
        ONLINE MERGE IS DISABLED    IN THIS PROCESSOR
        BUILD IS NOT SCHEDULED FOR NEXT IPL
        CPUID   ACTIVE   USING CURRENT DEFINITION
          B       YES              YES
          C       YES              YES
          D       NO               NO
          E       NO               NO
          Z       NO               NO
          0       NO               NO
        DISPLAY COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the online merge function.

ZNOPL STATUS–Display SNA Resource Status

Use this command to display information about the SNA resource definitions and the dynamic load process.

Requirements and Restrictions

None.

Format

```
►►──ZNOPL STatus──────────────────────────────────────────────────────►◄
```

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNOPL HELP
ZNOPL ?

Examples

The following information is displayed in the examples.

- Status of the RRT sections:

  CURRENT
  : Contains the current SNA network definitions. These definitions are used to define the SNA network when an initial program load (IPL) is performed on the TPF system.

  NEW/ALTERNATE
  : Contains the new SNA network definitions for the TPF system after you enter the ZNOPL LOAD command (regardless of whether you specify the FRESH or DYNAMIC parameter).

    Contains the old (or previous) SNA network definitions for the TPF system after you enter the ZNOPL UPDATE command for a dynamic load.

  UNUSED
  : Indicates that no SNA network definitions are currently loaded in the RRT section. The next time you perform a fresh or dynamic load, the SNA network definitions will be loaded to this RRT section.

- Whether the RRT was loaded using the dynamic load function or the fresh load function.
- Date and time that the offline ACF/SNA table generation (OSTG) SNA network definitions were created.
- A description of the OSTG SNA resource definitions.

  Note: The description is specified using the DESC= parameter in the PARM field of the JCL EXEC statement for the OSTG input data sets.

- Status of the ZNOPL UPDATE command:

  ENABLED
  : Can be entered to continue the dynamic load function. After you enter the

ZNOPL UPDATE command, the new (alternate) RRT becomes the current RRT and the current RRT becomes the new (alternate) RRT.

DISABLED
: Cannot be entered at this time.

IN PROGRESS
: Already entered and is in progress.

- Status of the ZNOPL FALLBACK command:

  ENABLED
  : Can be entered to reload the SNA network definitions from the new (alternate) RRT.

  DISABLED
  : Cannot be entered at this time.

  IN PROGRESS
  : Already entered and is in progress.

- Status of the ZNOPL MERGE command:

  ENABLED
  : Can be entered to complete the dynamic load function by updating the RVT with the current RRT definitions.

  DISABLED
  : Cannot be entered at this time.

  IN PROGRESS
  : Already entered and is in progress.

- Status of the ZNOPL BUILD command:

  SCHEDULED FOR NEXT IPL
  : RVT will be rebuilt during the next initial program load (IPL) using the current RRT.

  NOT SCHEDULED FOR NEXT IPL
  : RVT will ***not*** be rebuilt during the next IPL.

- Status of the processors in the TPF system:

  CPUID
  : ID of the processor.

  ACTIVE
  : Indicates if the processor is active.

  USING CURRENT DEFINITION
  : Indicates if the processor is using the latest RRT definitions.

Information about the SNA resource definitions is displayed in the following example.

Notice that a fresh load was just performed using section 2 of the RRT. Because a fresh load was just performed, there is no new (alternate) RRT. Therefore, section 1 of the RRT is unused. Also notice that because a fresh load was performed, the ZNOPL UPDATE, ZNOPL FALLBACK, and ZNOPL MERGE commands are disabled.

Two processors are active in the TPF system, and the new SNA resource definitions were incorporated on both of these processors.

```
User:   ZNOPL STATUS

System: NOPL0002I 14.34.25 SNA RESOURCE DEFINITION STATUS
        RRT SECTION 1 IS UNUSED
        RRT SECTION 2 IS CURRENT
        FRESH       LOADED
        DATE CREATED 10/11/95
        TIME CREATED 16.41.47
        DESCRIPTION TEST DEFINITIONS IN OSTGM2

        UPDATE IS DISABLED
        FALLBACK IS DISABLED
        ONLINE MERGE IS DISABLED    IN THIS PROCESSOR
        BUILD IS NOT SCHEDULED FOR NEXT IPL
        CPUID   ACTIVE   USING CURRENT DEFINITION
          B       YES              YES
          C       YES              YES
          D       NO               NO
          E       NO               NO
          Z       NO               NO
          0       NO               NO
        DISPLAY COMPLETE
```

In the following example, information about the SNA resource definitions is displayed after a dynamic load was performed.

Notice that section 1 of the RRT is now the current RRT and section 2 is now the new (alternate) RRT. None of the processors are using the new resource definitions in the current RRT. The ZNOPL MERGE command is enabled on this processor, which indicates that you must enter this command to incorporate the new resource definitions on this processor.

The ZNOPL FALLBACK command is also enabled. This indicates that you can fall back to the old SNA resource definitions, which are contained in section 2 of the RRT.

```
User:   ZNOPL STATUS

System: NOPL0002I 17.53.20 SNA RESOURCE DEFINITION STATUS
        RRT SECTION 1 IS CURRENT
        DYNAMICALLY LOADED
        DATE CREATED 10/14/95
        TIME CREATED 11.26.23
        DESCRIPTION OSTGM1

        RRT SECTION 2 IS NEW/ALTERNATE
        FRESH       LOADED
        DATE CREATED 10/11/95
        TIME CREATED 16.41.47
        DESCRIPTION TEST DEFINITIONS IN OSTGM2

        UPDATE IS DISABLED
        FALLBACK IS ENABLED
        ONLINE MERGE IS ENABLED     IN THIS PROCESSOR
        BUILD IS NOT SCHEDULED FOR NEXT IPL
        CPUID   ACTIVE   USING CURRENT DEFINITION
          B       YES              NO
          C       YES              NO
          D       NO               NO
          E       NO               NO
          Z       NO               NO
          0       NO               NO
        DISPLAY COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the fresh load function, dynamic load function, and falling back to previous resource definitions.

ZNOPL UPDATE–Update Dynamic Load

Use this command to update the resource resolution table (RRT) definitions after a dynamic load. The alternate RRT definitions become the current RRT definitions, and the previous current RRT definitions become the new alternate RRT definitions for fallback purposes.

Requirements and Restrictions

You can enter this command only after you enter the ZNOPL LOAD command with the DYNAMIC parameter.

Format

```
►►──ZNOPL UPdate──────────────────────────────────────────────────────────►◄
```

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNOPL HELP
  ZNOPL ?
- You can enter this command from any processor in the loosely coupled TPF system.
- A message is displayed to describe any errors that occur during the update function. After you correct the errors, you must do the following:
  – Load the offline ACF/SNA table generation (OSTG) input data set to the TPF system.
  – Enter the ZNOPL LOAD command with the DYNAMIC parameter.
  – Enter the ZNOPL UPDATE command.

Examples

The RRT definitions are updated in the following example. Notice that after you enter the ZNOPL UPDATE command, you must enter the ZNOPL MERGE command to update the resource vector table (RVT) and complete the dynamic load function.

```
User:   ZNOPL LOAD DYNAMIC B

System: NOPL0061A 12.20.51 SUCCESSFUL DYNAMIC LOAD, UPDATE ENABLED.
        COTC0087A 12.20.51 TCLS HPN    REMOVE SNA FROM DEVICE 424
                           VSN A00737  NOBLK
User:   ZNOPL UPDATE

System: NOPL0062A 12.27.39 UPDATE COMPLETE ON CPU-B , ENTER ZNOPL MERGE TO INCORPORATE
        NEW DEFINITIONS.
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the dynamic load process.

ZNPIU–Display PIU Trace Table

Use this command to display the entries in the path information unit (PIU) trace table.

Requirements and Restrictions

- Enter the ZNTRP command to start the PIU trace facility and specify the data that you want to trace before you enter this command.
- You can enter the ZNPIU command only in 1052 state or higher.

Format

```
►►──ZNPIU──────────────────────────────────────────────────────────────────►

    ┌─ ──COMPACT─┐     ──ALL──────┐    ┌─ ──NOControl─┐
►───┴────────────┴─── ─┬─────────┬┴────┴──────────────┴─────────────────────────────►◄
                       └─number──┘     └─ ──Control───┘
    │                                   ┌─ ──NOControl─┐  ┌─ ──NOHeaders─┐  ┌─ ──Data───┐  │
    └───┬─ ──FORMAT──┬─── ──┬─ALL────┬──┴──────────────┴──┴──────────────┴──┴───────────┴──┘
        └─ ──LONG────┘      └─number─┘  └─ ──Control───┘  └─ ──Headers───┘  └─ ──NOData─┘
```

COMPACT
creates a compacted display of the PIU trace table entries.

FORMAT
creates a formatted display of the PIU trace table entries. The transmission header (TH) values in the PIU trace table entries are translated. The entire response/request unit (RU) being traced is displayed.

LONG
creates the same display as the FORMAT parameter.

ALL
displays all the entries in the PIU trace table.

number
is the number of entries in the PIU trace table that you want to display, from 1–999.

Control
includes high-performance routing (HPR) control messages in the information that is displayed.

NOControl
omits HPR control messages from the information that is displayed.

Headers
includes the network layer header (NHDR) and transport header (THDR) in the network layer packets (NLPs) that are displayed.

NOHeaders
omits the NHDR and THDR from the NLPs that are displayed.

Data
includes the RU in the information that is displayed.

NOData
omits the RU from the information that is displayed.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNPIU HELP
  ZNPIU ?
- If you try to display more entries than are currently contained in the PIU trace table, all of the entries in the PIU trace table are displayed.
- The number of entries in the PIU trace table is shown in the last line of information that is displayed. This number includes HPR control messages, even if they are not included in the information that is displayed; that is, the NOCONTROL parameter was specified.
- Enter the ZNTRP command to specify the number of bytes of the RU to store in the PIU trace table.

Examples

The following list provides descriptions of the fields contained in the information that is displayed. A subset of these fields is displayed depending on the parameters that you specify.

BSN
: is the byte sequence number (BSN) of the rapid transport protocol (RTP) connection (NLP only)

CCW
: is the channel command word (CCW) area index

CNT
: is the count; that is, the combined length of the request/response header (RH) and the RU (FID4 only)

CTC HDR
: is the eight-byte channel-to-channel header (CTC only)

DNA
: is the destination network address (FID4 only)

DNAME
: is the destination name

  Note: The DNAME and ONAME fields contain the words ROUTE.SETUP if the entry contains an HPR ROUTE_SETUP command.

DRID
: is the destination resource identifier (RID)

  Note: For TPF/APPC sessions, either the DRID or ORID is the SCBID.

ERVR
: is the explicit route, virtual route, and transmission priority (FID4 only)

FID5
: is the FID5 transmission header (NLP only)

IN
: is the channel command word area index

LNKHDR
: is the link header (FID2 and NLP only)

NHDR
: is the network layer header (NLP only)

ONA
    is the origin network address (FID4 only)

ONAME
    is the origin name

Note: The DNAME and ONAME fields contain the words ROUTE.SETUP if the entry contains an HPR ROUTE_SETUP command.

ORID
    is the origin RID

Note: For TPF/APPC sessions, either the DRID or ORID is the SCBID.

PCID
    is the procedure correlation identifier

RH
    is the request/response header

RH INDICS
    is one or more of the following RH indicators:

−RSP
    Negative response

BB
    Begin brackets

CD
    Change direction

CEB
    Conditional end bracket

CS
    Code selection indicator

DFC
    Data flow control PIU

DR
    Definite response requested

EB
    End brackets

ED
    Enciphered data

ER
    Exception response requested

FI
    Format indicator

FIC
    First in chain

FMD
    Function management data PIU

LIC
    Last in chain

MIC
    Middle in chain

NC
: Network control PIU

NR
: No response requested

OIC
: Only in chain

PAC
: Pacing request or pacing response

PID
: Padded data

PRSP
: Positive response

QR
: Queued response

REQ
: Request

RLW
: Request larger window size

RSP
: Response

SC
: Session control PIU

SDI
: Sense data indicator

RTP
: is the rapid transport protocol control block (RTPCB) index (NLP only)

RU
: is the request/response unit or 8-byte channel-to-channel (CTC) header

RW
: is the read/write operation code

RWI
: is the read/write operation code

SA1
: is the session address (SA) that the TPF system assigned to the HPR LU-LU session (NLP only)

SA2
: is the SA that the remote RTP endpoint assigned to the HPR LU-LU session (NLP only)

SEGMENTS
: indicates the presence of the following optional segments in the THDR (NLP only):

ARB
: Adaptive Rate Based segment (X'22')

CF
: Connection Fault segment (X'12')

CIE
Connection Identifier Exchange segment (X'10')

COB
Client Out of Band segment (X'0F')

CS
Connection Setup segment (X'0D')

SI Switching Information segment (X'14')

STATUS
Status segment (X'0E')

SEQ
is the sequence number of the session

SID
is the session identifier (FID2 only)

TCID1
is the transport connection identifier (TCID) that the TPF system assigned to the RTP connection (NLP only)

TCID2
is the TCID that the remote RTP endpoint assigned to the RTP connection (NLP only)

TH
is bytes 0 and 1 of the FID2 transmission header (FID2 only)

THDR
is the transport header (NLP only)

TIME
is the time stamp

VRSQ
is the virtual route sequence number (FID4 only).

In the following example, 29 entries from the PIU trace table are displayed in a compacted format. The entries contain both FID2 and FID4 PIUs.

```
User:   ZNPIU 29

System: NPIU0004I 08.36.49 PIU TRACE TABLE
        RW IN  DRID   ORID   DNA     ONA  VRSQ SEQ  CNT    RH   RU
        RW IN  DRID   ORID   LNKHDR   TH  SID  SEQ         RH   RU
        RW IN  DRID   ORID   RTP         PCID       SEQ    RH   RU
        32 01 000000 000000 00790000 2F00 0101 0944      6B8000 31010703302000020
        31 01 000136 000070 00790000 2F00 0101 0944      EB8000 31010703302000020
        52 01 000070 000136 000E0000 2F00 0101 0945      6B8000 A0
        51 01 000136 000070 000E0000 2F00 0101 0945      EB8000 A0
        31 01 000136 000070 002E0000 2E00 0101 0001      039100 D3D401C3D3E4C2C2F
        32 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        51 01 000136 000070 002E0000 2E00 0101 0002      039100 D3D401C3D3E4C2C2F
        52 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        31 01 000136 000070 002E0000 2E00 0101 0003      039100 D3D401C3D3E4C2C2F
        32 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        51 01 000136 000070 00220000 2E00 0101 0004      039100 D3D404C3D3E4C2C2F
        52 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        31 01 000136 000070 00220000 2E00 0101 0005      039100 D3D404C3D3E4C2C2F
        32 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        51 01 000136 000070 00220000 2E00 0101 0006      039100 D3D404C3D3E4C2C2F
        52 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        32 02 000002 00000F 0B0000 1F0000 0000 0000 0018 2B0000 0F000001000000001
        31 02 00000F 000002 1F0000 0B0000 0000 0001 0018 2B0000 0F000001000000000
        51 02 00000F 000002 1F0000 0B0000 0000 0001 0028 2B0000 0B000001000001000
        52 02 000002 00000F 0B0000 1F0000 0000 0001 0034 2B0000 0C000001000101000
        31 02 00000F 000002 1F0000 0B0000 0000 0001 0016 2B8000 0D000001008000800
        32 02 000002 00000F 0B0000 1F0000 4000 0001 0004 AB8000 0D
        51 02 00000F 000002 1F0000 0B0000 4000 0001 0000
        31 01 000136 000070 00220000 2E00 0101 0007      039100 D3D404C3D3E4C2C2F
        32 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        51 01 000136 000070 002E0000 2E00 0101 0008      039100 D3D401C3D3E4C2C2F
        52 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        31 01 000136 000070 002E0000 2E00 0101 0009      039100 D3D401C3D3E4C2C2F
        32 01 000070 000136 00100000 2F00 0101 0000      830100 000001
        153 PIUS IN TRACE TABLE
```

In the following example, five entries from the PIU trace table are displayed in a compacted format. The entries contain both FID4 PIUs and 8-byte CTC headers.

```
User:   ZNPIU COMPACT 5

System: NPIU0004I 17.36.50 PIU TRACE TABLE
        RW IN  DRID   ORID   DNA     ONA  VRSQ SEQ  CNT    RH   RU
        RW IN  DRID   ORID   LNKHDR   TH  SID  SEQ         RH   RU
        RW IN  DRID   ORID   RTP         PCID       SEQ    RH   RU
        06 0B 00003F 800002 0A007C 0B0002 000C 0001 004D EB8000 31001307B0B0D0B13
        05 0B                                                   003F0001055E0544
        05 0B 000002 000001 0B0000 0A0001 000D 0003 0025 0B8000 8186460A0A688B508
        06 0B                                                   002000010544055E
        06 0B 000001 000002 0A0000 0B0001 000D 0003 0006 8B8000 818646
        35 PIUS IN TRACE TABLE
```

In the following example, five entries from the PIU trace table are formatted and displayed. The TH values are translated and the entire RU being traced is displayed. The entries contain FID2 PIUs, FID4 PIUs, and 8-byte CTC headers.

```
User:   ZNPIU FORMAT 5

System: NPIU0005I 09.52.59 PIU TRACE TABLE
        RWI-31 CCW-01 DNAME- ELMNGR            ONAME- VTAMNET.CLUAA001
           PCID-E383BE9553F2DA86 TIME-52.32  DRID-000102 ORID-0000BC
           LNKHDR-002E0000  TH-2E00 SID-0101 SEQ-0001   RH-039100
           RU-                 RH INDICS- REQ FMD OIC ER PAC
              0    0    D3D401C3 D3E4C1C1 F0F0F120 E3D7C6C1     LM.CLUAA 001.TPFA
             16   10    40404040 01000000 00E3D7C6 C1C1F3F0     ........ .TPFAA30
             32   20    40                                      .
        RWI-06 CCW-0B CTC HDR-003A00010205024D
        RWI-06 CCW-0B DNAME- TPFA              ONAME- TPFB
           PCID-0000000000000000 TIME-54.24  DRID-000001 ORID-000002
           DNA-0A0000 ONA-0B0001 ERVR-0001 VRSQ-0005 SEQ-0001 CNT-0020 RH-EB8000
           RU-ACTCDRM          RH INDICS- RSP SC FI PRSP
              0    0    14021111 40404040 40404040 05000000     ........ ........
             16   10    000B3F06 0600274E B40020FE 00           ........ .....
        RWI-05 CCW-0B CTC HDR-00670001052E03F4
        RWI-05 CCW-0B DNAME- APPC              ONAME- APPA
           PCID-0A0A3CC050800001 TIME-56.23  DRID-800001 ORID-00003F
           DNA-0B0002 ONA-0A007C ERVR-0001 VRSQ-000C SEQ-0001 CNT-004D RH-6B8000
           RU-BIND             RH INDICS- REQ SC FI OIC DR
              0    0    31001307 B0B0D0B1 3F3FA7F8 3F3F0602     ........ ...8....
             16   10    00000000 00000000 27000008 C1D7D7C1     ........ ....APPA
             32   20    40404040 1B000902 C6D4F140 40404040     ........ FM1.....
             48   30    0903000A 3CC05080 00010504 C1D7D7C1     ........ ....APPA
             64   40    0008C1D7 D7C34040 4040                  ..APPC.. ..
        99 PIUS IN TRACE TABLE
```

In the following example, a formatted NLP is displayed. The HEADERS parameter was specified so that the NHDR and THDR are also displayed.

```
User:   ZNPIU FORMAT 1 HEADERS

System: NPIU0005I 01.06.31 PIU TRACE TABLE
        RWI-31 CCW-01 DNAME- VTAMNET.LCL8DD    ONAME- G623
           PCID-E383BE95191C9BAA TIME-03.39  DRID-00023E ORID-00091A
           LNKHDR-00890000 TCID1-2366C462C2000002 TCID2-0BBEA50900000083 SEQ-0006
           RTP-000002 SA1-E383BE95191C9BAA SA2-0000000007000001 BSN-000002B4
           NHDR
              0    0    C2088000 0220D000 00000000 0000FF00     B....... ........
           THDR           SEGMENTS- STATUS
              0    0    0BBEA509 00000083 3004000A 0000004D     ........ ........
             16   10    000002B4 050E0000 00020E21 000000D3     ........ .......L
             32   20    00000000 00000000                       ........
           FID5-5C000006000000007000001      RH-038000
           RU-                 RH INDICS- REQ FMD OIC DR
              0    0    F1F7115C F0D8D9E3 F260D9D6 E4E3C9D5     17..0QRT 2-ROUTIN
             16   10    C740C9D5 C440C9D5 E5C1D3C9 C46B40C5     G.IND.IN VALID,.E
             32   20    D5E3C5D9 406F40C6 D6D940C8 C5D3D740     NTER...F OR.HELP.
             48   30    40404040 40404040 40404040 404E         ........ ......
        426 PIUS IN TRACE TABLE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the PIU trace facility and high-performance routing (HPR) support.

ZNPOL START–Start Polling an SNA Resource

Use this command to start polling a 37*x*5 (NCP or ALS) resource or a channel-to-channel (CTC) resource again after polling was stopped using the ZNPOL STOP command.

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNPOL START── ──sda──────────────────────────────────────────►◄
```

sda
is the 1- to 4-digit hexadecimal symbolic device address of a 37*x*5 or CTC resource.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNPOL HELP
  ZNPOL ?
- If an NCP resource is channel-attached to the TPF system, polling automatically starts for that NCP resource when you enter the ZNETW ACT command.
- Polling automatically starts for CTC resources following exchange ID (XID) processing.
- Polling starts for ALS resources when XID processing is started from VTAM.

Examples

Polling is started for the specified resource in the following example.

```
User:   ZNPOL START 001E

System: NPOL0001I 13.11.13  001E STARTED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNPOL STOP–Stop Polling an SNA Resource

Use this command to stop polling a Network Control Program (NCP), adjacent link station (ALS), or channel-to-channel (CTC) resource.

Requirements and Restrictions

None.

Format

```
►►──ZNPOL STOP── ──sda──────────────────────────────────────────────►◄
```

sda
: is the 1- to 4-digit hexadecimal symbolic device address of a 37*x*5 or CTC resource.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNPOL HELP
ZNPOL ?

Examples

Polling is stopped for the specified resource in the following example.

```
User:   ZNPOL STOP 001E

System: NPOL0005I 13.11.13  001E STOPPED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNPRG–Purge OMT Queue

Use this command to remove output messages that were queued on file by the output message transmitter (OMT) program and save them to a general tape.

Requirements and Restrictions

None.

Format

```
▶▶──ZNPRG── ──nodename──────────────────────────────────────────────▶◀
```

nodename
is a 1- to 17-character node name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNPRG HELP
  ZNPRG ?
- You can also remove the output messages by specifying a line number, interchange address, and terminal address (LNIATA) or a logical end-point identifier (LEID). See “ZPLMT–Purge LMT/OMT Queue” on page 1115 for more information.

Examples

The output messages for the specified node are removed from the queue and saved to a general tape in the following example.

```
User:   ZNPRG FA0306

System: NPRG0002I 09.17.57  QUEUE FOR FA0306        PURGE TO TAPE PLM, MSG CT - 7
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNRPT–Repeat Last OMT Message

Use this command to resend to a printer the last message that was queued on file by the output message transmitter (OMT) program.

Requirements and Restrictions

None.

Format

```
►►──ZNRPT── ──nodename──────────────────────────────────────────────►◄
```

nodename
is a 1- to 17-character node name of the printer. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRPT HELP
  ZNRPT ?
- You can also repeat the last output message by specifying a line number, interchange address, and terminal address (LNIATA) or a logical end-point identifier (LEID). See "ZRLMT–Repeat Last LMT/OMT Message" on page 1220 for more information.

Examples

In the following example, the last message that was queued on file by the OMT program is sent to the specified printer again.

```
User:   ZNRPT FA0302

System: NRPT0001I 09.13.57  MESSAGE REPEATED TO FA0302
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

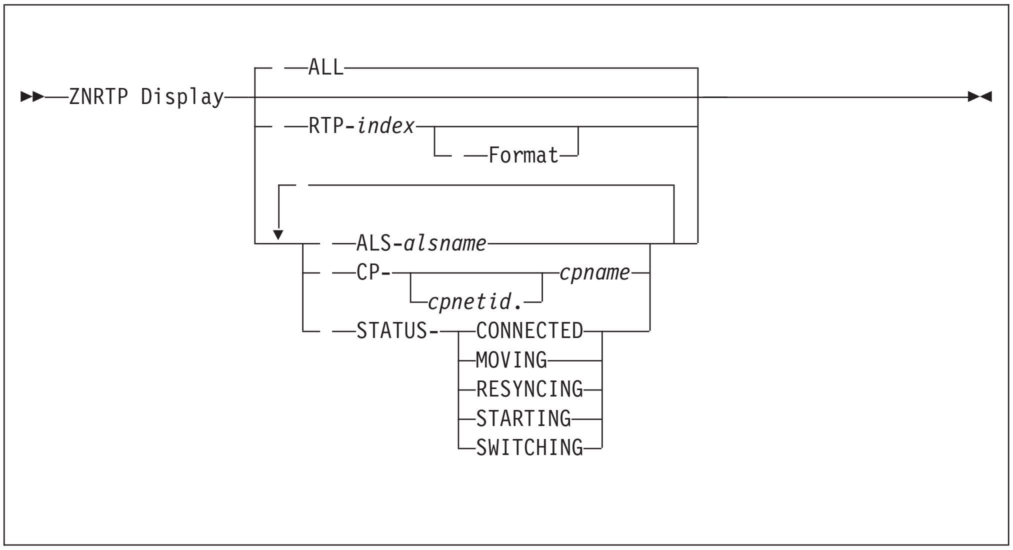
ZNRTP DISPLAY–Display Information about RTP Connections

Use this command to display information about rapid transport protocol (RTP) connections.

Requirements and Restrictions

The TPF system must have completed Systems Network Architecture (SNA) restart.

Format



ALL
displays information about all of the RTP connections in the TPF system.

ALS-*alsname*
displays information about all of the RTP connections in the TPF system that go through the specified adjacent link station (ALS), where *alsname* is the 1- to 8-character name of the ALS. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

CP-*cpnetid.cpname*
displays information about all of the RTP connections between the TPF system and the specified remote control point (CP), where *cpnetid.cpname* is a 1- to 17-character alphanumeric CP name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

RTP-*index*
displays the contents of the rapid transport protocol control block (RTPCB) entry with the specified RTPCB index, where *index* is a 1- to 6-character hexadecimal RTPCB index.

Format
formats and displays the RTPCB entry.

Note: The RTP connection must be active.

STATUS
displays information about all of the RTP connections that are in one of the following specified states:

CONNECTED
displays the RTP connections that are in their normal state with traffic flowing.

MOVING
displays the RTP connections that are performing a path switch because a TPF operator entered the ZNRTP SWITCH command.

RESYNCING
displays the RTP connections that are performing the RTP connection resynchronization process.

STARTING
displays the RTP connections that are starting.

SWITCHING
displays the RTP connections that are performing a path switch because of a failure in the high-performance routing (HPR) network.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNRTP HELP
ZNRTP ?

Examples

Information about all of the RTP connections is displayed in the following example, where:

RTP
is the RTPCB index.

CPNETID
is the CP network identifier of the remote RTP endpoint.

CPNAME
is the CP name of the remote RTP endpoint.

STATUS
is the status of the RTP connection, which can be one of the following:

CONNECTED
indicates that the RTP connection is in its normal state with traffic flowing.

MOVING
indicates that a path switch is in progress for the RTP connection because a TPF operator entered the ZNRTP SWITCH command.

RESYNCING
: indicates that the RTP connection is performing the RTP connection resynchronization process.

STARTING
: indicates that the RTP connection is starting.

SWITCHING
: indicates that a path switch is in progress for the RTP connection because of a failure in the HPR network.

COS
: is the class of service (COS) name used by the RTP connection.

ALS
: is the name of the ALS, if any, through which the RTP connection is currently routed.

SESSIONS
: is the number of HPR LU-LU sessions using the RTP connection.

SENDQ
: is the number of output messages that are queued and waiting to be sent out on the RTP connection.

UNACK
: is the number of network layer packets (NLPs) that the TPF system has sent out on the RTP connection that have not been acknowledged by the remote RTP endpoint.

```
User:   ZNRTP DISPLAY ALL

System: NRTP0001I 13.02.56 BEGIN ZNRTP DISPLAY

           RTP  CPNETID   CPNAME   STATUS     COS      ALS    SESSIONS SENDQ UNACK
          ----- -------- -------- --------- -------- -------- -------- ----- -----
          00002 VTAMNET  TPFVTAM3 CONNECTED INTERACT P30CA1B       13     6     4
          0000D VTAMNET  VTAM2    CONNECTED BATCH    P34CA6C        2     0     0
          00015 VTAMNET  TPFVTAM3 SWITCHING BATCH                   5     3     1
          0002A VTAMNET  TPFVTAM3 STARTING  BATCH    P30CA1B        5     0     0
       END OF ZNRTP DISPLAY
```

The specified RTPCB entry is displayed in the following example.

```
User:   ZNRTP DISPLAY RTP-1

System: NRTP0003I 13.04.23 RTPCB ENTRY CONTENTS
         03B0D000- 00000102 02810000 1DEB1508 C2000001 .....a.. ....B...
         03B0D010- 20E73C1B 00000313 E5E3C1D4 D5C5E340 .X...... VTAMNET
         03B0D020- E5E3C1D4 F2404040 D0000000 00000000 VTAM2    ........
         03B0D030- 7BC3D6D5 D5C5C3E3 000009AA C2ED9946 #CONNECT ....B.r.
         03B0D040- 09000001 00000000 00000000 00000070 ........ ........
         03B0D050- 00F00101 0012011C 0DE5E3C1 D4D5C5E3 .0...... .VTAMNET
         03B0D060- 4BE5E3C1 D4F20000 00000000 00000000 .VTAM2.. ........
         03B0D1A0- 00000000 00000000 0D800002 00D00000 ........ ........
         03B0D1B0- 00000000 00FF0000 00000000 00000000 ........ ........
         03B0D210- 00000000 00000000 B020E73C 00000000 ........ ..X.....
         03B0D240- 000001EB 0000007A 00020013 80000000 .......: ........
         03B0D290- 00000000 00000001 00000000 00000000 ........ ........
         03B0D2A0- 00000000 00000002 000001EB 0000007A ........ .......:
         03B0D2B0- 00000400 0000007E 00000000 0000013D .......= ........
         03B0D2D0- 00000078 00000000 0004004E 7A2E300B ........ ....:...
         03B0D2E0- 00000000 00000014 000002D1 00000000 ........ ...J....
         03B0D2F0- 00000000 00000000 8D8B8EE2 00000000 ........ ...S....
         03B0D310- 00000000 00000000 00000000 00000000 ........ ........
        END OF ZNRTP DISPLAY
```

The specified RTPCB entry is formatted and displayed in the following example, where:

CPNETID
: is the CP network identifier of the remote RTP endpoint.

CPNAME
: is the CP name of the remote RTP endpoint.

COS
: is the class of service (COS) name used by the RTP connection.

ALS
: is the name of the ALS, if any, through which the RTP connection is currently routed.

LINKSIZE
: is the maximum size for an NLP that can be sent over the RTP connection across its current route.

SESSIONS
: is the number of HPR LU-LU sessions using the RTP connection.

BSN SENT
: is the byte sequence number (BSN) of the next NLP that the TPF system will send on the RTP connection.

SENDQSIZ
: is the number of output messages that are queued and waiting to be sent out on the RTP connection.

INPUTQ
: is the number of input messages that are queued for the RTP connection because they were received out of order.

PIUTRACE
: is the status of the path information unit (PIU) trace facility for the RTP connection, which can be:

  NO
  : indicates that the PIU trace facility is not active for the RTP connection.

  STATCHNG
  : indicates that the PIU trace facility is active for only HPR state changes on the RTP connection. State changes include HPR ROUTE_SETUP commands, RTP connections starting, RTP connections stopping, and path switches.

  YES
  : indicates that the PIU trace facility is active for the RTP connection.

STATUS
: is the status of the RTP connection, which can be one of the following:

  CONNECTED
  : indicates that the RTP connection is in its normal state with traffic flowing.

  MOVING
  : indicates that a path switch is in progress for the RTP connection because a TPF operator entered the ZNRTP SWITCH command.

  RESYNCING
  : indicates that the RTP connection is performing the RTP connection resynchronization process.

STARTING
indicates that the RTP connection is starting.

SWITCHING
indicates that a path switch is in progress for the RTP connection because of a failure in the HPR network.

LOCAL TCID
is the transport connection identifier (TCID) that the TPF system assigned to the RTP connection.

REMOTE TCID
is the TCID that the remote RTP endpoint assigned to the RTP connection.

REMOTE NCE
is the network connection endpoint (NCE) of the remote RTP endpoint.

PATH SWITCHES
is the number of path switches that completed successfully for the RTP connection.

RETRANSMITS
is the number of NLPs that the TPF system retransmitted to the remote RTP endpoint on the RTP connection.

BSN RECEIVED
is the byte sequence number (BSN) of the next NLP that the TPF system is expecting to receive on the RTP connection.

UNACKNOWLEDGED
is the number of NLPs that the TPF system has sent out on the RTP connection that have not been acknowledged by the remote RTP endpoint.

PATH SW TIMER
indicates whether the path switch timer is running for the RTP connection and, if so, how many seconds remain for the path switch to be completed before the path switch will time out. If the path switch timer is not running, the time out value is displayed.

AVG ROUND TRIP
is the average number of seconds that it takes for the TPF system to send an NLP to the remote RTP endpoint and receive an acknowledgement on the RTP connection.

ALLOWED SEND RATE
is the number of bytes that the TPF system is allowed to send on the RTP connection in one second.

AVERAGE BYTES SENT
is the average number of bytes that the TPF system sent on the RTP connection in one second.

MAXIMUM BYTES SENT
is the maximum number of bytes that the TPF system sent on the RTP connection in one second.

SYNC
is the current synchronization number for the RTP connection.

ECHO
is the current echo number for the RTP connection.

AVERAGE BYTES RECEIVED
is the average number of bytes that the TPF system received on the RTP connection in 1 second.

MAXIMUM BYTES RECEIVED
is the maximum number of bytes that the TPF system received on the RTP connection in 1 second.

```
User:   ZNRTP DISPLAY RTP-1 FORMAT

System: NRTP0004I 13.34.23 RTPCB ENTRY CONTENTS FORMATTED

          CPNETID  - VTAMNET            STATUS - CONNECTED
          CPNAME   - A604           LOCAL TCID - 0D12124E C2000002
          COS      - #CONNECT      REMOTE TCID - 0C1D840A C3000002
          ALS      - P42CA1B        REMOTE NCE - DA
          LINKSIZE -     1452    PATH SWITCHES -        0
          SESSIONS -        1      RETRANSMITS -        0
          BSN SENT - 00009DAA     BSN RECEIVED - 00009DC1
          SENDQSIZ -        0   UNACKNOWLEDGED -        1
          INPUTQ   -        0    PATH SW TIMER -       90 STOPPED
          PIUTRACE - YES        AVG ROUND TRIP -        0.037602

          ALLOWED SEND RATE  -    37600   SYNC - 0001   ECHO - 0001
          AVERAGE BYTES SENT -     1622   AVERAGE BYTES RECEIVED -     1625
          MAXIMUM BYTES SENT -     3870   MAXIMUM BYTES RECEIVED -     3870

END OF ZNRTP DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about RTP connections and the RTPCB table.

ZNRTP HPR–Display or Change HPR Status

Use this command to enable high-performance routing (HPR) support, disable HPR support, or display the status of HPR support in the TPF system.

Requirements and Restrictions

- The TPF system must have completed Systems Network Architecture (SNA) restart.
- The TPF system must be running in Advanced Peer-to-Peer Networking (APPN) mode to enable HPR support. Enter **ZNAPN APPN** to place the TPF system in APPN mode.
- When you enable HPR support for the TPF complex, it is enabled only on the processors that have the rapid transport protocol control block (RTPCB) table and high-performance routing session address table (HPRSAT) defined.

Format

```
                  ┌─ ──Status──┐
►►──ZNRTP Hpr─────┼────────────┼──────────────────────────────────►◄
                  ├─ ──Disable─┤
                  └─ ──Enable──┘
```

Status
: displays the current status of HPR support for the TPF complex and for the processor on which the command was entered.

Disable
: disables HPR support. The TPF system will run with base APPN support.

Enable
: enables HPR support in the TPF system.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRTP HELP
  ZNRTP ?
- When the TPF system is first generated, HPR support is disabled. You must enter **ZNRTP HPR ENABLE** to start using HPR support.
- When you change the status of HPR support, the change is made on every TPF processor in the loosely coupled complex.
- If adjacent link station (ALS) links are already active when HPR support is enabled or disabled, the links must be deactivated and then activated again to reflect the change.

Examples

HPR support is enabled in the TPF system in the following example.

```
User:   ZNRTP HPR ENABLE

System: NRTP0005I 14.39.45 HPR SUPPORT IS NOW ENABLED
```

The status of HPR support in the TPF system is displayed in the following example.

```
User:   ZNRTP HPR

System: NRTP0007I 13.34.49 HPR SUPPORT IS ENABLED  FOR THE COMPLEX
                           HPR SUPPORT IS ENABLED  FOR THIS PROCESSOR
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about HPR support.

ZNRTP INACT–Deactivate RTP Connections

Use this command to deactivate rapid transport protocol (RTP) connections.

Requirements and Restrictions

You can enter this command only in CRAS state or higher.

Format

```
►►──ZNRTP INACT── ─┬─ALL───────┬──────────────────────────────────────►◄
                   └─RTP-index─┘
```

ALL
deactivates all of the RTP connections that are currently active on this processor.

RTP-*index*
deactivates the RTP connection with the specified rapid transport protocol control block (RTPCB) index, where *index* is a 1- to 6-character hexadecimal RTPCB index.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRTP HELP
  ZNRTP ?
- This command causes RTP connections to be unconditionally and immediately deactivated.
- Include the ZNRTP INACT command in the process that you use to deactivate a TPF processor.

Examples

All RTP connections are deactivated in the following example.

```
User:   ZNRTP INACT ALL

System: NRTP0011I 08.05.14 DEACTIVATION OF ALL RTP CONNECTIONS COMPLETED
```

The specified RTP connection is deactivated in the following example.

```
User:   ZNRTP INACT RTP-2A

System: NRTP0012I 08.05.28 RTP CONNECTION-00002A  CP-VTAMNET.VTAM2
                           DEACTIVATION COMPLETED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about RTP connections.

ZNRTP INITIALIZE–Initialize RTPCB Entries

Use this command to initialize the entries in the rapid transport protocol control block (RTPCB) table.

Requirements and Restrictions

- Do not use this command in a production environment because it resets Systems Network Architecture (SNA) control blocks in the TPF system without informing the network. Inconsistencies will occur.
- The TPF system must have completed SNA restart.

Format

```
►►──ZNRTP INITialize── ──┬─ALL───────┬──────────────────────────────────►◄
                         └─RTP-index─┘
```

ALL
initializes all RTPCB entries.

RTP-*index*
initializes the RTPCB entry with the specified RTPCB index, where *index* is a 1- to 6-character hexadecimal RTPCB index.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRTP HELP
  ZNRTP ?
- If you initialize the RTPCB entry for an RTP connection that has active high-performance routing (HPR) LU-LU sessions, the TPF control blocks for those HPR LU-LU sessions are cleaned up without informing the network.

Examples

All of the RTPCB entries are initialized in the following example.

```
User:   ZNRTP INITIALIZE ALL

System: NRTP0015I 09.23.13 INITIALIZATION OF ALL RTPCB ENTRIES COMPLETED
```

The specified RTPCB entry is initialized in the following example.

```
User:   ZNRTP INITIALIZE RTP-2A

System: NRTP0016I 09.14.13 INITIALIZATION OF RTPCB 00002A COMPLETED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the RTPCB table.

ZNRTP ROUTE–Display the RTP Connection Route

Use this command to display the route of a rapid transport protocol (RTP) connection.

Each node (or hop) along the route is shown, beginning with the node adjacent to the TPF system. The last node shown is the remote RTP endpoint.

Requirements and Restrictions

- The TPF system must have completed Systems Network Architecture (SNA) restart.
- The RTP connection cannot be in STARTING or SWITCHING state.

Format

```
►►──ZNRTP Route── ──RTP-index──────────────────────────────────────────►◄
```

RTP-*index*
: displays the route of the RTP connection with the specified rapid transport protocol control block (RTPCB) index, where *index* is a 1- to 6-character hexadecimal RTPCB index.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRTP HELP
  ZNRTP ?
- Both Advanced Peer-to-Peer Networking (APPN) and high-performance routing (HPR) route information is displayed.

Examples

The route for the specified RTP connection is displayed in the following example, where:

COS
: is the class of service (COS) name.

ANRF
: is the list of automatic network routing (ANR) labels representing the HPR route from the TPF system to the remote RTP endpoint.

HOP
: is the relative hop number of the node along the route.

CPNETID
: is the control point (CP) network identifier of the node.

CPNAME
: is the CP name of the node.

TG
: is the transmission group (TG) number of the link to this node.

```
User:   ZNRTP ROUTE RTP-34

System: NRTP0018I 15.55.36 RTP CONNECTION ROUTE INFORMATION

          COS - SNASVCMG
         ANRF - C3001EC0 021ADAFF

         HOP  CPNETID   CPNAME    TG
         ---  --------  --------  ---
           1  VTAMNET   VTAM2      21
           2  VTAMNET   A603       21
        END OF ZNRTP ROUTE DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about RTP connections.

ZNRTP SUMMARY–Display RTP Summary Information

Use this command to display summary information about rapid transport protocol (RTP) connections and the resources that they use.

You can also use this command to reset the maximum values for the summary information to the current values.

Requirements and Restrictions

The TPF system must have completed Systems Network Architecture (SNA) restart.

Format

```
►►──ZNRTP SUMmary──┬──────────┬──────────────────────────────────────►◄
                   └──Clear───┘
```

Clear
resets the maximum values to the current values.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZNRTP HELP
ZNRTP ?

Examples

RTP summary information is displayed in the following example, where:

CURRENT
is the current value of the statistic.

MAXIMUM
is the maximum value reached for the statistic.

MAXDATE
is the date when the maximum value was reached.

MAXTIME
is the time when the maximum value was reached.

CONNECTED
is the number of RTP connections that are in their normal state with traffic flowing.

MOVING
is the number of RTP connections that are performing a path switch because a TPF operator entered the ZNRTP SWITCH command.

RESYNCING
is the number of RTP connections performing the RTP connection resynchronization process.

STARTING
is the number of RTP connections that are starting.

SWITCHING
is the number of RTP connections that are performing a path switch because of a failure in the high-performance routing (HPR) network.

```
User:   ZNRTP SUMMARY

System: NRTP0002I 13.02.56 BEGIN ZNRTP SUMMARY
                                             CURRENT   MAXIMUM  MAXDATE MAXTIME
          NUMBER OF ACTIVE RTP CONNECTIONS       4        13    21DEC   08.16.06
          NUMBER OF ACTIVE HPR SESSIONS         25       123    21DEC   09.09.23
          PERCENTAGE OF THE HPRMT IN USE        21        54    22DEC   13.13.02

          STATUS OF RTP CONNECTIONS BY STATE

            STARTING  -    0
            CONNECTED -    3
            MOVING    -    0
            SWITCHING -    1
            RESYNCING -    0

        END OF ZNRTP SUMMARY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about RTP connections.

ZNRTP SWITCH–Switch an RTP Connection

Use this command to change the route of a rapid transport protocol (RTP) connection by starting a path switch.

Requirements and Restrictions

- You can enter this command only in NORM state or higher.
- CP-CP sessions must be active.

Format

```
►►──ZNRTP SWitch── ──RTP-index──────────────────────────────────────────►◄
```

RTP-*index*
: changes the route of the specified RTP connection, where *index* is the 1- to 6-character hexadecimal rapid transport protocol control block (RTPCB) index of the RTP connection.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRTP HELP
  ZNRTP ?
- After the path switch is completed successfully, use the ZNRTP ROUTE command to display the new route for the RTP connection.
- If a network failure occurs that causes an RTP connection to be automatically switched to a backup (less desirable) route, use this command to switch back to the primary route once that route becomes available again.
- Data traffic continues to flow on the RTP connection while the path switch is in progress.
- If the path switch fails, the RTP connection will continue to use its existing route.

Examples

The route for the specified RTP connection is switched in the following example.

```
User:   ZNRTP SWITCH RTP-34

System: NRTP0020I 08.05.13 RTP CONNECTION-000034  CP-VTAMNET.VTAM2
                           PATH SWITCH STARTED
        NRTP0021I 08.05.25 RTP CONNECTION-000034  CP-VTAMNET.VTAM2
                           PATH SWITCH COMPLETED
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about RTP connections and path switches.

ZNRVT–Display RVT Addresses and Contents

Use this command to display the contents of the resource vector table (RVT) for a particular node.

Requirements and Restrictions

Enter this command in 1052 state or higher.

Format

```
              ┌─ ──Addr──┐
►►──ZNRVT──┬──┼──────────┼─ ──┬─────────┬──nodename───────┬──────────►◄
           │  └─ ──MODES─┘    └─netid.──┘                 │
           └─ ──┬─DU──  ──┬─Rid── ──rid──────────────────┤
                └─DF──┘   └─Nodename── ──┬─────────┬──nodename─┘
                                         └─netid.──┘
```

Addr
produces an unformatted display of the RVT contents for the specified logical unit (LU).

MODES
displays all of the LU 6.2 modes that are associated with the specified remote LU. This parameter is valid only if the remote LU is in session with a TPF Advanced Program-to-Program Communications (TPF/APPC) LU.

netid.nodename
is the 1- to 17-character network qualified node name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

DU
produces an unformatted display of the RVT contents for the specified LU.

DF
produces a formatted display of the RVT contents for the specified LU. The DF parameter is valid only if the display is requested for a resource that is defined as an LU 6.2 resource. If the resource is not defined as an LU 6.2 resource, an unformatted display is produced.

Rid *rid*
displays the RVT contents for the specified resource identifier (RID), where *rid* is the RID.

Nodename *netid.nodename*
displays the RVT contents for the specified node, where *netid.nodename* is the 1- to 17-character network qualified node name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNRVT HELP
ZNRVT ?

Examples

An unformatted display of the RVT is produced for the specified node in the following example.

```
User:   ZNRVT NETA.CISBD002

System: NRVT0099I 14.53.22  RVT CONTENTS
        RVT1  CISBD002 IN NETA
         0092D4A0- D5C5E3C1 40404040 C3C9E2C2 C4F0F0F2 NETA     CISBD002
         0092D4B0- 00008000 00000000 00000000 00000000 ........ ........
         0092D4C0- 23078800 00160100 80850000 00000000 ..h..... .e......
         0092D4D0- 0D0D0000 00000000 00000000 00000000 ........ ........
         0092D4F0- 00000000 00000000 00370000 00000000 ........ ........
         0092D510- 40404040 40404040 40404040 40404040
         0092D520- 40404040 40404040 40404040 40404040
        RVT2  CISBD002 IN NETA
         00967400- 00000000 00000000 00000000 00000000 ........ ........
         00967410- 00000000 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

An unformatted display of the RVT contents is produced for the specified RID in the following example.

```
User:   ZNRVT DU R 00FC

System: NRVT0099I 09.17.57  RVT CONTENTS
        RVT1      L6G2B001
         00F876C0- 40404040 40404040 D3F6C7F2 C2F0F0F1            L6G2B001
         00F876D0- 00008000 00000000 00000000 00000000  ........  ........
         00F876E0- 69140C70 00160100 80000000 0000004E  ........  ........
         00F876F0- 0B0B0000 00000000 00000000 00000000  ........  ........
         00F87700- 00000000 00000000 00000000 00000000  ........  ........
         00F87710- 00000000 00000000 00000000 00000000  ........  ........
         00F87720- 00000000 00000000 00000000 00000000  ........  ........
         00F87730- 40404040 40404040 40404040 40404040
         00F87740- 40404040 40404040 40404040 40404040
         00F87750- 00000000 00000000 00000000 00000000  ........  ........
        RVT2      L6G2B001
         00FBD4E0- 00000000 00000000 00000000 00000000  ........  ........
         00FBD4F0- 00000000 00000000 00000000 00000000  ........  ........
```

A formatted display of the RVT contents is produced for the specified RID in the following example, where:

NET ID
: is the network ID

LUNAME
: is the LU name

RID
: is the hexadecimal resource ID of the partner logical unit

RVT1
: is the address of RVT section 1 for this session

RVT2
: is the address of RVT section 2 for this session

LUTYPE
: indicates whether the LU is a primary LU (PLU) or a secondary LU (SLU)

SCBID
: is the session control block ID of the first SCB chained off this RVT entry

SCBADDR
: is the address of the first SCB chained off this RVT entry

```
User:   ZNRVT DF R 0146

System: NRVT0020I 09.27.47  RVT CONTENTS FORMATTED
        NET ID            LUNAME  PBPB         RID     000146
        RVT1    00F4DAC0   RVT2    00FA8BC0
        LUTYPE  PLU        SCBID   00000000     SCBADR  00000000

        LU IS A CONTENTION WINNER    YES   LOCAL IS FIRST SPEAKER    NO
        BEGIN BRACKET PENDING        NO    BID REJ (0814 SENT/RECV)  NO
        SESSION IN BRACKET STATE     NO    RQD2 RESP PENDING TO REM  NO

        RQD1 RESP PENDING            NO    RQD2 RESP PENDING         NO
        DATA PURGE IN PROGRESS       NO    SEQ. NUMBER WRAP PENDING  NO
        LOCAL IN SEND STATE          NO    CHANGE DIRECTION SENT     NO
```

Information about all of the sessions associated with the specified remote LU is displayed in the following example, where:

MODENAME
: is the mode name.

PARTNER
: is the name of the partner (local) LU.

LIMIT
: is the session limit for the mode name.

WINNERS
: is the minimum limit of contention-winner sessions for the mode name.

LOSERS
: is the minimum limit of contention-loser sessions for the mode name.

ALOC
: is the number of sessions presently allocated to conversations.

ACT
: is the number of active and bound sessions at this time.

AVAIL
: is the number of available sessions at this time. If this number is negative, is represents the number of sessions that need to be deactivated to reach the new lower session limits.

```
User:   ZNRVT MODES APPA

System: NRVT0030I 09.22.16 MODES
        MODENAME  PARTNER   LIMIT WINNERS  LOSERS     ALOC    ACT  AVAIL
        -----------------------------------------------------------------
        SNASVCMG  PBPB          2       1       1        0      1      1
        NORLINS   PBPB          7       2       2        2      2      5
        HURICANE  PBPB          8       0       0        1      8      0
        GRATEFUL  PBPB        121      75      13        0      6    115
        END OF MODENAME DISPLAY
```

Information about secondary LU (SLU) threads that are in session with the specified remote LU is displayed in the following example.

```
User:   ZNRVT MODES APPA

System: NRVT0030I 09.24.59 MODES
        MODENAME  PARTNER   LIMIT WINNERS  LOSERS     ALOC    ACT  AVAIL
        -----------------------------------------------------------------
        TPFLU62   PBPBB001      1       0       0        0      1      0
        TPFLU62   PBPBB002      1       0       0        1      1      0
        TPFLU62   PBPBB003      1       0       0        0      0      1
        TPFLU62   PBPBB006      1       0       0        0      0      1
        TPFLU62   PBPBB013      1       0       0        0      1      0
        END OF MODENAME DISPLAY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about TPF Advanced Program-to-Program Communications (TPF/APPC).

ZNRVT INITIALIZE–Initialize RVT

Use this command to initialize an entry in the resource vector table (RVT) for a particular node.

Requirements and Restrictions

Do not use this command in a production environment because it resets the SNA control block indicators to their initial state, which is inactive, without changing their status in the network. Inconsistencies will occur.

Format

```
►►──ZNRVT INITialize── ─┬─────────┬─nodename──────────────────────►◄
                        └─netid.──┘
```

netid.nodename
is a 1- to 17-character network qualified node name. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNRVT HELP
  ZNRVT ?
- If you specify a remote LU that has TPF/APPC sessions, all associated session control blocks (SCBs) and conversation control blocks (CCBs) are also initialized.

Examples

In the following example, the entry in the RVT table for the specified node is initialized.

```
User:   ZNRVT INIT NETA.CISBD002

System: SSCP0029I 14.53.22     LU    CISBD002 ZNRVT INITIALIZATION COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNSCB–Display Session Control Block

Use this command to display the specified TPF Advanced Program-to-Program Communications (TPF/APPC) session control block (SCB).

Requirements and Restrictions

You can enter this command only in 1052 state or higher.

Format

```
►►──ZNSCB──┬─┬──────┬─ ──SCB-scbid──┬──────────────────────────────►◄
           │ └─ ──DF─┘               │
           └─ ──Sum──────────────────┘
```

DF
: produces a formatted display of the specified SCB. If you omit this parameter, the display is not formatted.

SCB-*scbid*
: is the session control block (SCB) ID of the LU-LU session that you want to display. The session control block identifier (SCBID) is in the format of the 3-byte resource identifier (RID).

Sum
: displays the number of active and inactive SCBs.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNSCB HELP
ZNSCB ?

Examples

A formatted display of the SCB data is produced in the following example, where:

SCBID
: is the session control block ID.

FORWARD CHAIN
: is the SCB forward chain.

MODE
: is the TPF/APPC mode name.

PCID
: is the procedure correlation ID.

RVTADR
: is the resource vector table address of the owning (remote) RVT.

OWNING RID
: is the resource identifier (RID) of the owning (remote) RVT.

PARTNER LU
: is the partner (local) LU name.

PARTNER RID
: is the hexadecimal RID of the partner LU.

CCBID
: is the conversation control block ID.

CCBADR
: is the address of the CCB.

```
User:   ZNSCB DF SCB-800002

System: NSCB0001I 17.33.24 SCB CONTENTS FORMATTED
        SCBID    800002             FORWARD CHAIN  00000000
        MODE     TPFMOD1
        PCID          .        .0A0A688B50800002
        RVTADR   025D5800           OWNING RID   000040
        PARTNER LU         .APPA     PARTNER RID  00003F
        CCBID    00000119           CCBADR       026BC2BC
        SESSION IN BRACKET STATE    YES   LOCAL IS FIRST SPEAKER    NO
        BEGIN BRACKET PENDING       NO    BID REJ (0814 SENT/RECV)  NO
        CHANGE DIRECTION SENT       NO    RQD2 RESP PENDING TO REM  NO
        RQD1 RESP PENDING           NO    RQD2 RESP PENDING         NO
        DATA PURGE IN PROGRESS      NO    SEQ. NUMBER WRAP PENDING  NO
        LOCAL IN SEND STATE         YES
```

In the following example, an unformatted display of the SCB data is produced.

```
User:   ZNSCB SCB-800002

System: NSCB0002I 17.34.20 SCB CONTENTS
        SCB1     SCBID-800002
         026C00F4- 40404040 40404040 C1D7D7C3 40404040          APPC
         026C0104- 00000000 C0000000 0A0A688B 50800002 ........ ........
         026C0114- 48140872 00160008 10F80000 00000002 ........ .8......
         026C0124- 0B0BA700 00003F00 003F0000 00000B00 ..x..... ........
         026C0134- 00000000 00000000 80000000 00000119 ........ ........
         026C0144- 00000000 000A007C 00800002 00000000 .......@ ........
         026C0164- 40404040 40404040 40404040 40404040
         026C0174- 40404040 40404040 40404040 40404040
         026C0194- 00000000 026C0014 E3D7C6D4 D6C4F140 ........ TPFMOD1
         026C01A4- 00050003 00020000 00010000 00000002 ........ ........
         026C01C4- 00160000 00400000 00000000 00000000 ..... .. ........
        SCB2
         026CE020- 00000000 00000000 00000000 80100000 ........ ........
         026CE030- 0000003F 00000000 00000000 00000000 ........ ........
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The number of active and inactive SCBs is displayed in the following example.

```
User:   ZNSCB SUM

System: NSCB0003I 17.27.58 SCB SUMMARY INFORMATION
        NUMBER OF ACTIVE SCBS              2
        NUMBER OF INACTIVE SCBS          248
        TOTAL NUMBER OF SCBS             250
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNSCB INITIALIZE–Initialize Session Control Block

Use this command to initialize TPF Advanced Program-to-Program Communications (TPF/APPC) session control blocks (SCBs).

Requirements and Restrictions

- You can enter this command only in NORM state.
- Use this command only in a test environment because it resets the SNA control block indicators to their initial state, which is inactive, without changing their status in the network. Inconsistencies will occur.

Format

```
►►──ZNSCB Initialize── ──┬─All────────┬─────────────────────────────────►◄
                         └─SCB-scbid──┘
```

All
initializes all the SCBs.

SCB-*scbid*
initializes the specified SCB, where *scbid* is the session control block identifier (SCBID) in the format of a 3-byte resource identifier (RID).

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNSCB HELP
  ZNSCB ?
- If you initialize an SCB that has an active conversation, the conversation control block (CCB) representing that conversation is also initialized.

Examples

The specified SCB is initialized in the following example.

```
User:   ZNSCB INIT SCB-800009

System: NSCB0010I 09.32.03 INITIALIZATION COMPLETE
```

All the SCBs are initialized in the following example.

```
User:   ZNSCB INIT ALL

System: NSCB0011I 14.05.45 INITIALIZATION OF ALL SCBS COMPLETED SUCCESSFULLY
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about SNA communications.

ZNSID–Side Information Table Function

Use this command to:

- Add an entry to the side information table
- Change an entry in the side information table
- Display an entry in the side information table
- Initialize the side information table
- Load additions and changes to the side information table
- Remove an entry from the side information table.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- The first time you bring up the TPF system, you must initialize the side information table before you can perform the other ZNSID functions.

Format

```
►►──ZNSID── ──────────────────────────────────────────────────────►

                                    ┌─ ──────────────────────┐
                                    ▼                        │
►──┬─┬─Add─────┬─ ──Name-symname── ─┬─Tp-tpname─────────────┬┴──────────►◄
   │ └─Change──┘                    ├─Lu-─┬─────────┬─luname─┤ │
   │                                │     └─netid.──┘        │ │
   │                                └─Mode-modename──────────┘ │
   ├─┬─Display─┬─ ──Name-symname───────────────────────────────┤
   │ └─Remove──┘                                               │
   ├─INITIALIZE────────────────────────────────────────────────┤
   └─LOAD── ─┬─Tape-tapename─┬─────────────────────────────────┘
             └─Gds-gdsname───┘
```

Add

adds an entry to the online side information table.

Note: Each entry that you add to the side information table must have a symbolic destination name (NAME), transaction program name (TP), partner LU (LU), and mode name (MODE). To accommodate long entries, the ADD parameter requires that you specify only one of the TP, LU, and MODE parameters. You can specify the other parameters by using the CHANGE parameter. See the examples for more information.

Change

changes an entry in the side information table. You can change the transaction program name (TP), partner LU (LU), or mode name (MODE).

Display

displays information about an entry in the side information table.

Remove

removes an entry from the online side information table.

INITIALIZE

initializes the side information table.

LOAD
loads entries from a side information data set that is generated offline. The offline data can be contained on either a TPF general tape or on a TPF general data set (GDS).

Loading an offline data set can cause entries to be added to or removed from the online side information table. If an entry to be added from the offline data set has the same symbolic destination name as an entry in the online side information table, the entry from the offline data set replaces the online entry.

See the *TPF ACF/SNA Data Communications Reference* for information about the offline side information program.

Name-*symname*
is a 1- to 8-character alphanumeric symbolic destination name for an entry in the side information table. This name is used as the key field for accessing online side information table entries to perform the ZNSID ADD, CHANGE, DISPLAY, and REMOVE functions.

Tp-*tpname*
is a 1- to 64-byte transaction program name for an entry in the side information table. This name can include letters, numbers, symbols, and unprintable characters.

To allow the entry of such names, the TP parameter can include bytes that are specified in hexadecimal. Use the dollar sign ($) as a delimiter for these hexadecimal substrings.

- Pairs of hexadecimal digits entered between 2 single dollar signs ($) are converted to the corresponding binary values; for example, T-$07F0$ sets the transaction program name of the specified entry to X'07F0'.
- Hexadecimal substrings can be freely intermixed with literal substrings in the same TP specification, as in T-A01BCD020304E05, which would set the transaction program name to X'C101C2C3C4020304C505'.
- If a dollar sign is actually desired in the transaction program name, 2 consecutive dollar signs ($$) should be used for each dollar sign needed. For example, T-X$$Y$$Z sets the transaction program name to "XYZ", and T-$$$$$$32$F1$$F0$$$ sets the transaction program name to "$$$3210$".
- The transaction program name cannot include blank characters (X'40').

Lu-*netid.luname*
is a 1- to 17-character partner LU for an entry in the side information table. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Mode-*modename*
is a 1- to 8-character mode name for an entry in the side information table. Mode names must begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify a mode name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the name. For example, if the mode name is $MODNAME, you must type **$$MODNAME**.

Tape-*tapename*
is the 3-character name of the tape that contains the offline data set.

Gds-*gdsname*
is the 1- to 16-character alphanumeric name of the GDS that contains the offline data set.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNSID HELP
  ZNSID ?
- All values for the TP, LU, and MODE parameters are verified for the correct length and format before they are entered in the online side information table. If you specify the length or format incorrectly, the appropriate response is sent to the console, followed by the parameter value in error and a pointer to the error. See the examples for more information.

Examples

The side information table is initialized in the following example. Once the side information table is initialized it is ready to accept new data and any previously existing data is destroyed.

```
User:   ZNSID INITIALIZE

System: NSID0004I 11.57.23 SIDE INFORMATION TABLE INITIALIZED
```

Changes and additions are loaded to the side information table from the TPF general tape named SID in the following example. Note that entries can also be read from a general data set.

Entries with new symbolic destination names are added to the table. Entries with duplicate names overwrite the old information.

```
User:   ZTMNT SID E46 A I

System: COTN0046I  TMNT 0123 - TAPE SID MOUNTED ON DEVICE E46 ...

User:   ZNSID LOAD TAPE-SID

System: NSID0037I 17.12.23 LOADING SIDE INFORMATION DATA
        NSID0017I 17.12.23 USER TEXT FROM SIDE INFORMATION TAPE
        NEW DESTINATIONS
        TAPE WAS CREATED 01/17/91
        --- END OF USER TEXT ---
        NSID0005I 17.12.24 NEW SIDE INFORMATION TABLE ENTRIES LOADED
        ENTRIES PROCESSED  -       1990
        ENTRIES ADDED      -       1901
        ENTRIES CHANGED    -          0
        ENTRIES REMOVED    -         89
        ERRORS DURING LOAD -          0
```

A new entry named PROG1 is added to the side information table in the following example. When this entry is used by a program, the ALLOCATE verb brings up a session using SLOW mode with a NET.LUA partner LU, which has a transaction program called INVENTORY.

```
User:   ZNSID ADD NAME-PROG1 TP-INVENTORY LU-NET.LUA MODE-SLOW

System: NSID0001I 14:03:05 NEW ENTRY ADDED TO SIDE INFORMATION TABLE
```

The partner LU for the PROG1 entry in the side information table is changed in the following example.

```
User:   ZNSID CHANGE NAME-PROG1 LU-NET.LUB

System: NSID0081I 14:10:03 SIDE INFORMATION ENTRY CHANGED
```

A long entry is added to the side information table in the following example. Only the transaction program name is specified for the ZNSID ADD command. The partner LU and mode name are specified using the ZNSID CHANGE command.

```
User:   ZNSID A N-REMOTE T-LONG$01$NAME$02$WITH$03$UNPRINTABLE$04$CHARACTERS

System: NSID0006A 12.34.56 NEW ENTRY ADDED TO SIDE INFORMATION TABLE, ENTRY IS
                           INCOMPLETE

User:   ZNSID C N-REMOTE L-NETNET.REMOTELU

System: NSID0019A 12.35.14 SIDE INFORMATION ENTRY CHANGED, ENTRY IS INCOMPLETE

User:   ZNSID C N-REMOTE M-FIRSTCLS

System: NSID0018I  12.35.43 SIDE INFORMATION ENTRY CHANGED
```

Entries in the side information table are displayed in the following example. If a transaction program name contains only letters, numbers, and symbols, it is displayed as text. If it contains unprintable characters, it is displayed as both hexadecimal digits and text.

```
User:   ZNSID DISPLAY NAME-PROG1

System: NSID0002I 14.27.06 SIDE INFORMATION ENTRY DISPLAY
           NAME- PROG1
             TP- INVENTORY
             LU- NET.LUB
           MODE- SLOW

User:   ZNSID D N-REMOTE

System: NSID0002I 14.27.17 SIDE INFORMATION ENTRY DISPLAY
             NAME- REMOTE
          TP(HEX)- D3D6D5C7 01D5C1D4 C502E6C9 E3C803E4 LONG.NAM E.WITH.U
                   D5D7D9C9 D5E3C1C2 D3C504C3 C8C1D9C1 NPRINTAB LE.CHARA
                   C3E3C5D9 E2                         CTERS
               LU- NETNET.REMOTELU
```

The PROG1 entry is removed from the side information table in the following example.

```
User:   ZNSID REMOVE NAME-PROG1

System: NSID0003I 14:15:12 SIDE INFORMATION ENTRY REMOVED
```

Incorrect values were specified for the TP and LU parameters in the following example. The pointer (---A) indicates where the error was found.

```
User:   ZNSID A N-XXX L-NETID2LONG.LUOK T-HEX$ABC$ERR M-MODEOK

System: NSID0024E 15.32.38 TP NAME CONTAINS HEX SUBSTRING WITH ODD NUMBER OF DIGITS
             HEX$ABC$ERR
        ------------A
        NSID0021E 15.32.38 NETWORK ID IS LONGER THAN 8 CHARACTERS
             NETID2LONG
        -------------A
```

In the following example, a side information offline data set that contains an incorrect entry is loaded to the TPF system. This results in an entry dump and the load ends.

Note: The TPF offline side information tape generation program does not include incorrect entries, such as the one in this example, in the generated side information data set.

```
User:   ZNSID LOAD GDS-MYGDS

System: NSID0013E 00.15.63 SIDE INFORMATION GDS ENTRY ERROR DUMP
        INVALID TP NAME
        INVALID LU NAME
        INVALID MODE
             NAME- C6D6D6C2 C1D95A5A                   FOOBAR..
        TP NAME LENGTH -  90
               TP- E6C8C1E3 40E6D6E4 D3C440C8 C1D7D7C5 WHAT WOU LD HAPPE
                   D540C9C6 40C1D540 D6C6C6D3 C9D5C540 N IF AN  OFFLINE
                   C4C1E3C1 40E2C5E3 40C9D5C3 D3E4C4C5 DATA SET  INCLUDE
                   C440C1D5 40C5D5E3 D9E840E2 C8C1E340 D AN ENT RY THAT
        LU NAME LENGTH - 195
               LU- D6D5E3C1 C9D5C5C4 40D1E4E2 E340D9C1 ONTAINED  JUST RA
                   D5                                  N
        MODE LENGTH    - 196
             MODE- D6D440C4 C1E3C16F                   OM DATA.
        OP CODE -   1 - ADD ENTRY

        NSID0014E 00.15.64 PARTIAL SIDE INFORMATION TABLE LOAD TERMINATED DUE TO ERROR
        ENTRIES PROCESSED  -         1
        ENTRIES ADDED      -         0
        ENTRIES CHANGED    -         0
        ENTRIES REMOVED    -         0
        ERRORS DURING LOAD -         1
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the side information table and TPF/APPC in general.

ZNSPA–Initialize SPA

Use this command to initialize the scratch pad area (SPA) records.

Requirements and Restrictions

You can enter this command only in 1052 state or higher. Otherwise, meaningless information is displayed.

Format

```
►►──ZNSPA── ──┬─ALL─────────────────────┬──────────────────────────────►◄
              ├─MOD── ──module──────────┤
              └─NOD── ──netid.nodename──┘
```

ALL
initializes all the SPA records.

MOD *module*
initializes all the SPA records on the specified module, where *module* is the module number.

NOD *netid.nodename*
initializes the SPA record for the specified node, where *netid.nodename* is the 1- to 17-character network qualified name of a network addressable unit (NAU). The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Note: To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZNSPA HELP
ZNSPA ?

Examples

The scratch pad area record for the specified node is initialized in the following example.

```
User:   ZNSPA NOD RBS1B005

System: NSPA0017I 09.34.31 START INITIALIZING SPA
        NSPA0001I 09.34.31 INITIALIZATION COMPLETE
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about scratch pad area (SPA) records.

ZNTRP–PIU Trace Facility

Use this command to do the following:

- Start or stop the path information unit (PIU) trace facility. You can specify the data that you want to trace. For example, you can trace the data for a specific:
  - Logical unit (LU)
  - Cross-domain resource manager (CDRM)
  - System services control point (SSCP)
  - Adjacent link station (ALS)
  - Channel-to-channel (CTC) link
  - Network Control Program (NCP)
  - Rapid transport protocol (RTP) connection.

  You can also specify that you want to trace:
  - Data transferred between the TPF system and all remote Systems Network Architecture (SNA) resources
  - Only network control commands, virtual route (VR) pacing requests, and VR pacing responses for CTC links or NCP resources
  - Only high-performance routing (HPR) traffic, which includes all network layer packets (NLPs) on all RTP connections and HPR ROUTE_SETUP commands
  - Only data flowing for high-performance routing (HPR) state changes.
- Start or stop writing the PIU trace table to the real-time tape.
- Specify the number of bytes of the request/response unit (RU) to store in the PIU trace table for data messages on LU-LU sessions.
- Specify the number of bytes of the RU to store in the PIU trace table for SNA commands, data flowing on CDRM-CDRM sessions, and data flowing on CP-CP sessions.
- Display the status of the PIU trace facility.
- Display information about the resources being traced.

Requirements and Restrictions

You can enter this message only in 1052 state or higher.

Format

```
►►──ZNTRP── ─────────────────────────────────────────────────────────────►

►──┬─STArt──┬─┬─────────┬─ ─ALL──────────────────────────────────┬──┬───►◄
   │        │ └─ ─Vronly─┘ └┬────────────────────┬─ ─Id-resource─┘  │  │
   │        │               └─ ─Netid-identifier─┘                  │  │
   │        ├─ ─Hpr──┬─────────────┬────────────────────────────────┤  │
   │        │        └─ ─Statchng─┘                                 │  │
   │        ├─ ─Rtp-index───────────────────────────────────────────┤  │
   │        └─ ─Tape────────────────────────────────────────────────┘  │
   ├─STOp──┬─ ─ALL──────────────────────────────────┬──────────────────┤
   │       ├─ ─ALLR─────────────────────────────────┤                  │
   │       ├┬────────────────────┬─ ─Id-resource────┤                  │
   │       │└─ ─Netid-identifier─┘                  │                  │
   │       ├─ ─Hpr──────────────────────────────────┤                  │
   │       ├─ ─Rtp-index────────────────────────────┤                  │
   │       └─ ─Tape─────────────────────────────────┘                  │
   ├─Rusz── ─rsize─────────────────────────────────────────────────────┤
   ├─Crusz── ─csize────────────────────────────────────────────────────┤
   └─Display─┬────────────┬────────────────────────────────────────────┘
             └─ ─Options─┘
```

STArt
starts the PIU trace facility or starts writing the PIU trace table to the real-time tape.

Vronly
traces only network control commands, VR pacing requests, and VR pacing responses.

Note: If you specify VRONLY, the value of the ID parameter must be the name of an NCP or CTC link.

ALL
traces data transferred between the TPF system and all remote SNA resources.

Netid-*identifier*
traces data for a specific LU, CDRM, SSCP, ALS, CTC link, or NCP, where *identifier* is the 1- to 8-character network identifier (ID) of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. You can use the asterisk (*) as a wildcard character to specify a group of network identifiers that begin or end with a common string of characters. For example, to start tracing data for all resources that have a network identifier beginning with SNA, enter **SNA***.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Id-*resource*
traces data for a specific LU, ALS, CDRM, SSCP, CTC link, or NCP, where *resource* is the 1- to 8-character name of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you specify the name of an NCP, ALS, or CTC link, all data for resources that have sessions through that NCP, ALS, or CTC link is traced.
2. You can use the asterisk (*) as a wildcard character to specify a group of resource names that begin or end with a common string of characters. For example, to start tracing data for all resources that have a name ending with NCP, enter ***NCP**.
3. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Hpr
traces all HPR traffic, which includes all NLPs on all RTP connections and HPR ROUTE_SETUP commands.

Statchng
traces only data flowing for HPR state changes on RTP connections. State changes include HPR ROUTE_SETUP commands, RTP connections starting, RTP connections stopping, and path switches.

Rtp-*index*
traces data on a specific RTP connection, where *index* is the 1- to 6-digit hexadecimal rapid transport protocol control block (RTPCB) index of the RTP connection.

Note: The specified RTP connection must be active.

Tape
automatically writes each 4-KB block of the PIU trace table to the real-time tape once the 4-KB block is full.

Note: If the length of the queue to the real-time tape exceeds the tape queue threshold value defined by the PIUTAPEQ parameter in the SNAKEY macro, the PIU trace facility stops writing the PIU trace table to tape until the queue falls below this value again.

STOp
stops the PIU trace facility or stops writing the PIU trace table to the real-time tape.

ALL
stops tracing data transferred between the TPF system and all remote SNA resources. (This trace was defined by entering ZNTRP START ALL.) Traces defined for specific resources remain active.

ALLR
stops the PIU trace facility completely. All of the traces defined end and data is no longer stored in the PIU trace table.

Netid-*identifier*
stops a trace defined for a specific LU, CDRM, SSCP, ALS, CTC link, or NCP, where *identifier* is the 1- to 8-character network identifier of the

resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. You can use the asterisk (*) as a wildcard character to specify a group of network identifiers that begin or end with a common string of characters. For example, to stop traces defined for all resources that have a network identifier beginning with SNA, enter **SNA***.
2. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Id-*resource*
stops a trace defined for a specific LU, CDRM, SSCP, ALS, CTC link, or NCP, where *identifier* is the 1- to 8-character network identifier of the resource. The network ID and name of a resource must both begin with a letter (A–Z), @, #, or $. The remaining characters can be letters (A–Z), numbers (0–9), @, #, or $.

Notes:

1. If you specify the name of an NCP, ALS, or CTC link, data for resources that have sessions through that NCP or ALS will no longer be traced, unless a specific trace was defined for one of those resources.
2. You can use the asterisk (*) as a wildcard character to specify a group of resource names that begin or end with a common string of characters. For example, to stop traces defined for all resources that have a name ending with NCP, enter ***NCP**.
3. To specify an SNA network ID or resource name that contains dollar signs ($), you must type two consecutive dollar signs ($$) for each dollar sign in the network ID or resource name. For example, if the SNA resource name is $SNANAME, you must type **$$SNANAME**.

Hpr
stops tracing HPR traffic. (This trace was defined by entering ZNTRP START HPR.) If traces were defined for specific LUs or RTP connections, those traces remains active.

Rtp-*index*
stops a trace defined for a specific RTP connection, where *index* is the 1- to 6-digit hexadecimal RTPCB index of the RTP connection.

Tape
stops automatically writing the PIU trace table to real-time tape. When you specify this parameter, the PIU trace table is written to tape one last time, whether or not the the current 4-KB block is full.

Rusz *rsize*
changes the number of bytes of the RU being stored in the PIU trace table for data messages on LU-LU sessions, where *rsize* is a decimal number in the range 8–3840.

Crusz *csize*
changes the number of bytes of the RU being stored in the PIU trace table for SNA commands, data flowing on CDRM-CDRM sessions, and data flowing on CP-CP sessions, where *csize* is a decimal number in the range 8–3840.

Display
displays information about the data being traced.

Options
displays the status of the PIU trace facility. The information displayed indicates:
- Whether the PIU trace table is currently being written to tape
- The size of the RU stored in the PIU trace table for data messages on LU-LU sessions
- The size of the RU stored in the PIU trace table for SNA commands, data flowing on CDRM-CDRM sessions, and data flowing on CP-CP sessions.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZNTRP HELP
  ZNTRP ?
- The PIU trace facility stores 21 bytes of the RU in the PIU trace table for data messages on LU-LU sessions unless you change this value by using the RUSZ parameter.
- The PIU trace facility stores 21 bytes of the RU in the PIU trace table for SNA commands, data flowing on CDRM-CDRM sessions, and data flowing on CP-CP sessions unless you change this value by using the CRUSZ parameter.
- The 8-byte header that precedes all read and write operations during normal CTC data transfer operations is also stored in the PIU trace table.
- If you specify the ID parameter without specifying the NETID parameter, the PIU trace facility selects the first resource in the collating sequence of the network identifiers.
- You can use the ZNDLU, ZNALS, and ZNETW DISPLAY commands to display status information about a specific LU, CDRM, SSCP, ALS, NCP, or CTC link.
- You can use the ZNPIU command to display the data that was traced and stored in the PIU trace table.

Examples

The PIU trace facility is started for the PBPB resource in the SNANET network in the following example.

```
User:   ZNTRP START NETID-SNANET ID-PBPB

System: ZNTRP0001I 13.11.13 PIU TRACE STARTED FOR SNANET.PBPB
```

The PIU trace facility is started for all resources that have a name ending in the characters AT in the following example.

```
User:   ZNTRP START ID-*AT

System: NTRP0008I 13.11.13  PIU TRACE STARTED FOR ALL RESOURCES CHOSEN WITH
        SELECTION CRITERIA --- NAME- *AT
```

In the following example, the PIU trace facility is started for all the resources defined in the TPF system.

```
User:   ZNTRP START ALL

System: NTRP0002I 14.02.21 PIU TRACE STARTED FOR ALL RESOURCES
```

In the following example, the ALL parameter is used to stop the PIU trace facility for all resources in the TPF system. Keep in mind that the PIU trace facility is not stopped for the PBPB resource or the resources that have a name ending with the characters AT because these resources were already being traced when ZNPIU START ALL was entered in the previous example.

```
User:   ZNTRP STOP ALL

System: NTRP0004I 14.05.41 PIU TRACE STOPPED FOR ALL RESOURCES
```

In the following example, the ALLR parameter is used to stop the PIU trace facility for all resources in the TPF system. In this case, the PIU trace facility is stopped for the PBPB resource and for the resources that have a name ending in the characters AT.

```
User:   ZNTRP STOP ALLR

System: NTRP0004I 14.07.31 PIU TRACE STOPPED FOR ALL RESOURCES
```

Information about the resources being traced is displayed in the following example.

```
User:   ZNTRP DISPLAY

System: NTRP0048I 12.31.25 PIU TRACE ACTIVE FOR THE FOLLOWING SELECTED RESOURCES
        NETID      NAME       RTP INDEX    DEVICE TYPE     VRONLY
        --------   --------   ---------   --------------   ------
                   N42E720                 ALS/NCP/CTC      YES
                   G623                         LU
                               000001     RTP CONNECTION
        END OF DISPLAY
```

The status of the PIU trace facility is displayed in the following example.

```
User:   ZNTRP DISPLAY OPTIONS

System: NTRP0045I 23.23.21 PIU TRACE OPTIONS INFORMATION
        PIU TRACE IS ACTIVE TO REAL TIME TAPE
        200 BYTES OF RU ARE BEING TRACED FOR USER DATA MESSAGES
        100 BYTES OF RU ARE BEING TRACED FOR SNA COMMANDS
        END OF DISPLAY
```

The size of the RU being stored in the PIU trace table for data messages is changed to 300 bytes in the following example.

```
User:   ZNTRP RUSZ 300

System: NTRP0013I 23.26.43 TRACING 300 BYTES OF RU FOR USER DATA MESSAGES
```

Related Information

See *TPF ACF/SNA Data Communications Reference* for more information about the PIU trace facility and RTP connections.

ZOLDR ACCEPT–Accept a Loadset

Use this command to accept a loadset (that is, replace the base program versions with the program versions in a specified loadset).

Requirements and Restrictions

- The specified loadset must be activated on all active processors in the TPF system and scheduled for activation on any processors in the TPF system that are not currently active.
- You cannot accept selectively activated loadsets or loadsets that contain unallocated programs.
- If programs contained in the specified loadset are also contained in other loadsets in the TPF system (that is, the specified loadset ***intersects*** with other loadsets), you must accept the loadsets in the same order in which they were activated. This ensures that the correct version of the program replaces the base program version.

Format

```
►►──ZOLDR ACCept── ──lsname──────────────────────────────────────────►◄
```

lsname
is the 5- to 8-character alphanumeric name of the loadset that you want to accept.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- After a loadset is accepted, it is deleted from the system when processors stop using it. Therefore, no E-type loader functions can be performed on the loadset after it is accepted.
- You cannot fall back to earlier versions of programs in a loadset once the loadset is accepted.

Examples

The CAPRES loadset is accepted in the following example.

```
User:   ZOLDR ACCEPT CAPRES

System: OLDR1016I 19.39.59 CELA - ACCEPT REQUEST RECEIVED
        OLDR2042I 19.39.59 ACCEPT OF LOADSET CAPRES SCHEDULED
         FOR CPU B
        OLDR2042I 19.39.59 ACCEPT OF LOADSET CAPRES SCHEDULED
         FOR CPU C
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET CAPRES SCHEDULED
         FOR CPU B
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET CAPRES SCHEDULED
         FOR CPU C
        OLDR3216I 19.39.59 COLO - LOADSET CAPRES ACCEPTED AND SCHEDULED FOR DEACTIVATION
        AND DELETION
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET CAPRES FOR CPU B COMPLETE AS REQUESTED
        BY CPU B
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET CAPRES FOR CPU C COMPLETE AS REQUESTED
        BY CPU B
        OLDR5252I 19.39.59 COLF - LOADSET CAPRES DELETED
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR ACTIVATE–Activate a Loadset

Use this command to activate a loadset on a specific processor or on all processors. When new entry control blocks (ECBs) are created, they use the programs contained in the activated loadset.

Requirements and Restrictions

None.

Format

```
►►──ZOLDR ACTivate── ──lsname──┬──────┬──┬───────┬──────────────────────►◄
                               └─ ──id─┘  └─ ──SEL─┘
```

lsname
is the 5- to 8-character alphanumeric name of the loadset that you want to activate.

id
is the 1-character alphanumeric CPU ID of the processor on which you want to activate the loadset. If you do not specify a CPU ID, the loadset is activated on all the processors in the complex where it is not currently active.

Notes:

1. If a processor is not currently active or if it is running with a different program base, the programs in the loadset are not used by that processor until you perform an initial program load (IPL) using the same program base as the processor from which the ZOLDR ACTIVATE command was entered.
2. This parameter is ignored on systems that were not generated for a shared data processing system (SDPS).

SEL
selectively activates the loadset. Selective activation allows you to control which ECBs have access to the loadset. For more information about selective activation, see *TPF System Installation Support Reference*.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- When a loadset is activated on a processor, all programs in that loadset become available simultaneously.

Examples

The CAPRES loadset is activated in the following example.

```
User:   ZOLDR ACTIVATE CAPRES

System: OLDR1016I 19.39.59 CELA - ACTIVATE REQUEST RECEIVED
        OLDR2036I 19.39.59 ACTIVATE OF LOADSET CAPRES SCHEDULED
         FOR CPU B
        OLDR2015I 19.39.59  CEL2 - LOADSET CAPRES ASSIGNED ACTIVATION NUMBER 14 ON CPU B
        OLDR2043I 19.39.59 ACTIVATE OF LOADSET CAPRES COMPLETED
         FOR CPU B  AS REQUESTED BY CPU B
```

The TEST1 loadset is selectively activated in the following example.

```
User:   ZOLDR ACT TEST1 SEL

System: OLDR1016I 19.39.59 CELA - ACTIVATE REQUEST RECEIVED
        OLDR2037I 19.39.59 SELECTIVE ACTIVATE OF LOADSET TEST1 SCHEDULED FOR CPU B
        OLDR2037I 19.39.59 SELECTIVE ACTIVATE OF LOADSET TEST1 SCHEDULED FOR CPU C
        OLDR2015I 19.39.59 LOADSET TEST1 ASSIGNED ACTIVATION NUMBER 1 ON CPU B
        OLDR2044I 19.39.59 SELECTIVE ACTIVATE OF LOADSET TEST1 COMPLETED
         FOR CPU B AS REQUESTED BY CPU B
        OLDR2044I 19.39.59 SELECTIVE ACTIVATE OF LOADSET TEST1 COMPLETED
         FOR CPU C AS REQUESTED BY CPU B
```

The LOADSTOP loadset is activated on CPU Z from CPU Z in the following example.

```
User:   ZOLDR ACT LOADSTOP Z

System: OLDR1016I 19.39.59 CELA - ACTIVATE REQUEST RECEIVED
        OLDR2036I 19.39.59 ACTIVATE OF LOADSET LOADSTOP SCHEDULED
         FOR CPU Z
        OLDR2015I 19.39.59 CEL2 - LOADSET LOADSTOP ASSIGNED ACTIVATION NUMBER 37 ON CPU Z
        OLDR2043I 19.39.59 ACTIVATE OF LOADSET LOADSTOP COMPLETED
         FOR CPU Z  AS REQUESTED BY CPU Z
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR ALTER–Alter E-Type Loader Values

Use this command to change the following E-type loader values:

- Extra program allocation table (PAT) slot threshold percentage
- E-type loader fixed file record threshold percentage
- Number of incompatible PAT slots reported during the E-type loader load function before the function is ended
- Number of mismatched program allocation characteristics reported during the E-type loader load function
- Time interval for starting the E-type loader police routine
- Time interval for starting the E-type loader long-running job detection routine and the E-type loader reclaim detection routine.

Requirements and Restrictions

None.

Format

```
►►──ZOLDR ALter── ──┬─Xpat── ──Threshold── ──pp──────────────┬──────────────────────►◄
                    ├─Record── ──Threshold── ──pp────────────┤
                    ├─PAt── ──Report── ──nn──────────────────┤
                    └─POlice── ──time── ──Detect── ──time────┘
```

Xpat Threshold *pp*
: changes the extra PAT slot threshold percentage, where *pp* is the percentage. When the number of available extra PAT slots falls below the specified percentage, the TPF system issues an attention message.

Record Threshold *pp*
: changes the E-type loader fixed file record threshold percentage, where *pp* is the percentage. When the number of E-type loader fixed file records falls below the specified percentage, the TPF system issues an attention message.

PAt Report *nn*
: changes the maximum number of incompatible PAT slots the system reports before ending the E-type loader load function and the maximum number of mismatched program allocation characteristics the system reports during the E-type loader load function, where *nn* is a decimal value from 1–99.

POlice *time*
: changes the time interval for starting the E-type loader police routine, where *time* is a time interval from 1–999 seconds.

Detect *time*
: changes the time interval for starting the E-type loader long-running job detection routine and the E-type loader reclaim detection routine, where *time* is a time interval from 1–999 seconds.

Note: The time interval for the E-type loader long-running job detection routine and the E-type loader reclaim detection routine must be a multiple of the time interval for the police routine.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- All changes take effect immediately.
- The values changed using the ZOLDR ALTER command affect all subsystems.

Examples

The E-type loader fixed file record threshold percentage is changed to 5% in the following example.

```
User:   ZOLDR ALTER RECORD THRESHOLD 5

System: OLDR1016I 19.39.59 ALTER REQUEST RECEIVED
        OLDR5007I 19.39.59 CLDT - VALUE ALTERED SUCCESSFULLY
           OLD VALUE - 0
           NEW VALUE - 5
           THE NEW VALUE TAKES EFFECT IMMEDIATELY
```

In the following example, the time interval for starting the E-type loader police routine is changed to 2 seconds. The time interval for starting the E-type loader long-running job detection routine and the E-type loader reclaim detection routine is changed to 4 seconds.

```
User:   ZOLDR ALTER POLICE 2 DETECT 4

System: OLDR1016I 19.39.59 CELA - ALTER REQUEST RECEIVED
        OLDR5005I 19.39.59 CLDT - VALUES ALTERED SUCCESSFULLY
           OLD VALUES - 10 AND 20
           NEW VALUES - 2 AND 4
           THE NEW VALUES TAKE EFFECT IMMEDIATELY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR ALTER PROGCHAR–Alter E-Type Loader Rules

Use this command to change the allocation characteristics assigned to unallocated programs when the unallocated programs are loaded and activated using the E-type loader.

Requirements and Restrictions

The parameters for this command must be entered in the specified order.

Format

```
►►──ZOLDR ALter PROGChar── ─────────────────────────────────────────────────────►
                          ├─ ──CR─┤  ├─ ──SHared──────┤  ├─ ──24bit─┤
                          └─ ──FR─┘  ├─ ──COmmon──────┤  └─ ──31bit─┘
                                     ├─ ──UNprotected─┤
                                     ├─ ──ISunique────┤
                                     └─ ──PRivate─────┘

►─────────────────────────────────────────────────────────────────────────────►◄
  ├─ ──Key0───┤  ├─ ──Montc───┤  ├─ ──Restrict───┤  ├─ ──Cmb───┤
  └─ ──NOKey0─┘  └─ ──NOMontc─┘  └─ ──NORestrict─┘  └─ ──NOCmb─┘
```

CR
: specifies that unallocated programs are treated as core resident programs.

FR
: specifies that unallocated programs are treated as file resident programs.

SHared
: specifies that unallocated programs are treated as shared programs. That is, all ECBs share the same key-protected storage copy of the unallocated program, but not all ECBs see the unallocated program at the same address.

  Both core resident and file resident programs can be defined as shared programs.

COmmon
: specifies that unallocated programs are treated as common programs. That is, all ECBs share the same key-protected storage copy of the unallocated program, and all ECBs see the unallocated program at the same address.

  Both core resident and file resident programs can be defined as common programs.

UNprotected
: specifies that unallocated programs are treated as unprotected programs. That is, all ECBs share the same unprotected storage copy of the unallocated program, and all ECBs see the unallocated program at the same address. This parameter is intended for self-modifying programs.

  Only file resident programs can be defined as unprotected programs.

ISunique
: specifies that unallocated programs are treated as I-stream unique programs. That is, all ECBs using an unallocated program on a specific I-stream share the

same unprotected storage copy of the program, and all ECBs see the unallocated program at the same address. This parameter is intended for self-modifying, I-stream unique programs.

Only file resident programs can be defined as I-stream unique programs.

PRivate
specifies that unallocated programs are treated as private programs. That is, an ECB is supplied with a new storage copy of an unallocated program each time the ECB enters the unallocated program.

Only file resident programs can be defined as private programs.

24bit
specifies that unallocated programs are entered in 24-bit mode.

31bit
specifies that unallocated programs are entered in 31-bit mode.

Key0
specifies that unallocated programs can change the protection key to 0.

NOKey0
specifies that unallocated programs cannot change the protection key to 0.

Montc
specifies that unallocated programs can enter supervisor state.

NOMontc
specifies that unallocated programs cannot enter supervisor state.

Restrict
specifies that unallocated programs can issue restricted macros.

NORestrict
specifies that unallocated programs cannot issue restricted macros.

Cmb
specifies that unallocated programs can obtain common storage blocks.

NOCmb
specifies that unallocated programs cannot obtain common storage blocks.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- All changes take effect immediately. The next time an unallocated program is activated, it assumes the new allocation characteristics.
- If you activate a loadset that contains an unallocated program, and that unallocated program is also contained in another loadset that was activated ***before*** you changed the allocation characteristics using the ZOLDR ALTER command, the unallocated program is not assigned the new program allocation characteristics. Instead, it is assigned the allocation characteristics of the unallocated program that is contained in the other loadset.

  Therefore, to ensure that unallocated programs are assigned the most recent allocation characteristics when you activate a loadset, deactivate all the other loadsets that contain a version of the unallocated program before you activate the new loadset.

- The values changed using the ZOLDR ALTER PROGCHAR command affect all subsystems.

Examples

The allocation characteristics assigned to unallocated programs are changed in the following example.

```
User:   ZOLDR ALTER PROGCHAR FR SHARED 24BIT NOKEY0 NOMONTC NORESTRICT NOCMB

System: OLDR1016I 19.39.59 CELA - ALTER REQUEST RECEIVED
        OLDR5006I 19.39.59 CLDT - VALUES ALTERED SUCCESSFULLY
           OLD VALUES - FR SHARED 31BIT NOKEY0 NOMONT NORESTRICT CMB
           NEW VALUES - FR SHARED 24BIT NOKEY0 NOMONT NORESTRICT NOCMB
           THE NEW VALUES TAKE EFFECT IMMEDIATELY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR CLEAR–Initialize E-Type Loader Structures

Use this command to clear and initialize all file resident E-type loader structures.

Requirements and Restrictions

- Enter this command only during the following situations:
  - When the first initial program load (IPL) of the system is being performed
  - When E-type loader records are damaged.
- You must specify the BP parameter the first time you enter this command. Otherwise, all of the file resident E-type loader structures may not be initialized.
- After you enter the ZOLDR CLEAR command, you must perform an IPL on the processor where you entered the command and on all processors that have the same program base.
- You cannot perform the E-type loader load, accept, or reclaim functions while the E-type loader clear function is in progress.
- The ZOLDR CLEAR command does not initialize the E-type loader rules database (ERD). To initialize the ERD, enter the ZOLDR ALTER command.
- The ZOLDR CLEAR command does not initialize the core resident E-type loader structures or the program allocation table (PAT). These resources are initialized when an IPL is performed on the processor where the command was entered and on the processors that have the same program base.

Format

```
►►──ZOLDR CLEAR── ──ALL──┬──────┬──────────────────────────────────────────►◄
                         └─ ──BP─┘
```

ALL
initializes all E-type loader structures.

BP
bypasses any error responses to the ZOLDR CLEAR command and continues to initialize the E-type loader records, regardless of whether E-type loader functions are being performed.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZOLDR HELP
ZOLDR ?

Examples

The E-type loader structures are cleared and initialized in the following example.

```
User:   ZOLDR CLEAR ALL

System: OLDR1016I 19.39.59 CELA - CLEAR REQUEST RECEIVED
        OLDR0911A 19.39.59 COL7 - E-TYPE LOADER CLEAR PROCESSING COMPLETED,
            THIS PROCESSOR MUST BE RE-IPLED IMMEDIATELY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR DEACTIVATE–Deactivate a Loadset

Use this command to deactivate a loadset on a specific processor or on all processors. When a loadset is deactivated, no new ECBs can use the loadset. However, if an ECB exists that can use the loadset, it continues to do so until the ECB exits from the TPF system.

Requirements and Restrictions

None.

Format

```
►►──ZOLDR DEActivate── ──lsname──┬──────────┬──┬──────────┬──────────────►◄
                                 ├─ ──id────┤  ├─ ──EXIt──┤
                                 └─ ──FORce─┘  └─ ──SERrc─┘
```

lsname
: is the 5- to 8-character alphanumeric name of the loadset you want to deactivate.

id
: is the 1-character alphanumeric CPU ID of the processor on which you want to deactivate the loadset. If you do not specify a CPU ID or the FORCE parameter, the loadset is deactivated on all the processors in the complex where it is currently active.

FORce
: immediately deactivates a loadset without checking the status of the loadset. Forced deactivation of a loadset occurs only on the processor from which this command was entered.

  Note: Deactivating a loadset without checking the status can cause unpredictable results. Therefore, it is recommended that you use the FORCE parameter only when the loadset is causing a critical situation and previous attempts to deactivate the loadset without specifying the FORCE parameter failed.

EXIt
: causes ECBs to exit if they attempt to enter a program in the loadset.

SERrc
: causes ECBs to exit and issue a system error dump if they attempt to enter a program in the loadset.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZOLDR HELP
ZOLDR ?

Examples

The HOTEL1 loadset is deactivated on every processor in the following example.

```
User:   ZOLDR DEACTIVATE HOTEL1

System: OLDR1016I 19.39.59 CELA - DEACTIVATE REQUEST RECEIVED
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET HOTEL1 SCHEDULED
         FOR CPU B
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET HOTEL1 SCHEDULED
         FOR CPU C
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET HOTEL1 SCHEDULED
         FOR CPU Z
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET HOTEL1 FOR CPU B COMPLETED AS REQUESTED
        BY CPU B
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET HOTEL1 FOR CPU C COMPLETED AS REQUESTED
        BY CPU C
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET HOTEL1 FOR CPU Z COMPLETED AS REQUESTED
        BY CPU Z
```

The APPL201 loadset is deactivated on CPU B in the following example.

```
User:   ZOLDR DEAC APPL201 B

System: OLDR1016I 19.39.59 CELA - DEACTIVATE REQUEST RECEIVED
        OLDR2038I 19.39.59 DEACTIVATE OF LOADSET APPL201 SCHEDULED
         FOR CPU B
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET APPL201 FOR CPU B COMPLETED AS REQUESTED
        BY CPU B
```

In the following example, the APPL201 loadset is deactivated on CPU B. ECBs using programs in the APPL201 loadset exit when they attempt to enter the programs in the loadset.

```
User:   ZOLDR DEAC APPL201 B EXIT

System: OLDR1016I 19.39.59 CELA - DEACTIVATE REQUEST RECEIVED
        OLDR2039I 19.39.59 DEACTIVATE EXIT OF LOADSET APPL201 SCHEDULED
         FOR CPU B
        OLDR1058I 19.39.59 CLE4 - DEACTIVATE LOADSET APPL201 FOR CPU B COMPLETED AS REQUESTED
        BY CPU B
```

The BETATEST loadset is deactivated without checking the status of the loadset in the following example.

```
User:   ZOLDR DEACT BETATEST FORCE

System: OLDR1016I 19.39.59 CELA - DEACTIVATE REQUEST RECEIVED
        OLDR1057I 19.39.59 COLA - DEACTIVATE LOADSET BETATEST FOR CPU B COMPLETE
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR DELETE–Delete a Loadset

Use this command to delete a loadset from the TPF system.

Requirements and Restrictions

The loadset cannot be active on any processor in the TPF system.

Format

```
►►──ZOLDR DELete── ──lsname──────────────────────────────────────────────►◄
```

lsname
is the 5- to 8-character alphanumeric name of the loadset that you want to delete.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- A loadset is deleted by changing the name of the loadset to a temporary name, which is generated by the TPF system, and then marking the loadset as *delete pending* until all active ECBs using the loadset exit. When this occurs, the loadset and the resources associated with that loadset are automatically deleted by the TPF system.

Examples

The LOAD01 loadset is deleted in the following example.

```
User:   ZOLDR DELETE LOAD01

System: OLDR1016I 19.39.59 CELA - DELETE REQUEST RECEIVED
        OLDR5207I 19.39.59 COLE - LOADSET LOAD01   IS NOW SCHEDULED FOR
                  DELETION
        OLDR5252I 19.39.59 COLF - LOADSET LOAD01 DELETED
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR DISPLAY–Display E-Type Loader Information

Use this command to display the following E-type loader information:

- Information about the loadsets loaded using the E-type loader
- Rules followed during E-type loader processes
- Values of variables used during E-type loader processes.

Requirements and Restrictions

None.

Format

```
►►─ZOLDR DIsplay── ──┬─ALl─────────────────────┬──────────────────────►◄
                     ├─ACtivate────────────────┤
                     │          ┌─.──────┐     │
                     │          ▼        │     │
                     ├─PRog── ───progname─┴────┤
                     │             ┌─.────┐    │
                     │             ▼      │    │
                     ├─Loadset── ───lsname─┴───┤
                     ├─Inter── ──lsname────────┤
                     ├─Rules───────────────────┤
                     └─Values──────────────────┘
```

ALl
: displays a list of all the loadsets and their status. If a loadset is active, a list of the processors where it is active is also displayed.

ACtivate
: displays a list of all the active loadsets and the processors on which they are active.

PRog
: displays a list of all the loadsets, either active or inactive, that contain one or more specified programs.

progname
: is a 4-character alphanumeric name of the program. You can use an asterisk (*) as a wildcard character for this parameter.

  Note: You can specify as many as 20 program names at one time.

Loadset
: displays the status of a loadset and a list of the programs contained in the loadset.

lsname
: is the 5- to 8-character alphanumeric name of a loadset. You can use an asterisk (*) as a wildcard character for the LOADSET parameter, but not for the INTER parameter.

  Note: You can specify as many as 20 loadset names at one time for the LOADSET parameter.

Inter
displays a list of the loadsets that *intersect* with a specific loadset. Loadsets intersect when they contain common programs.

Note: Only loadsets that were activated before the specified loadset are checked for intersections.

Rules
displays the rules that are used during E-type loader processes. These rules are defined using the ZOLDR ALTER command.

Values
displays the values of the variables used during E-type loader processes. These values are defined using the ZOLDR ALTER and the ZCTKA ALTER commands.

Note: The number of extra program allocation table (PAT) slots displayed is for the current initial program load (IPL). To view the number of extra PAT slots allocated for the next IPL, use the ZCTKA DISPLAY command.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- Information is displayed in alphabetic order unless system resources are not available to perform the necessary sort functions.
- The format of the time and dates displayed in the loadset information depends on your installation.

Examples

All the loadsets and their status are displayed in the following example.

```
User:   ZOLDR DISP ALL

System: OLDR1220I 17.25.41 CLE3 - BEGIN LOADSET INFORMATION
         LOADSET   STATUS
         CAPRES    CREATED ON 04/28/94 08.17.18
                   ACTIVATED ON CPUID C ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID D ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID E ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID Z ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID 0 ON 04/28/94 08.26.08

         LUAPPL    CREATED ON 04/28/94 08.17.18
                   INACTIVE

         TEST1     CREATED ON 04/28/94 08.06.28
                   INACTIVE
        CLE3 - END OF DISPLAY
```

All the active loadsets and the processors on which they are active are displayed in the following example.

```
User:   ZOLDR DISP ACT

System: OLDR1220I 17.25.41 CLE3 - BEGIN LOADSET INFORMATION
         LOADSET   STATUS
         CAPRES    CREATED ON 04/28/94 08.17.18
                   ACTIVATED ON CPUID C ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID D ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID E ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID Z ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID 0 ON 04/28/94 08.26.08

         TEST2     CREATED ON 04/28/94 08:06:28
                   ACTIVATED ON CPUID Z ON 04/28/94 08.28.23
        CLE3 - END OF DISPLAY
```

All the loadsets that contain programs that have names beginning with the letters RSD are displayed in the following example.

```
User:   ZOLDR DISP PROG RSD*

System: OLDR1240I 17.25.41 CLDW - BEGIN PROGRAM DISPLAY
          RSD* IS A MEMBER OF THESE LOADSETS -
          LOADSET           PROGRAM-VERSION

          TEST1             RSD7JN    RSD4JN

          TEST2             RSD7JN    RSD4JN
        CLDW - END OF DISPLAY
```

The status of the CAPRES loadset and a list of the programs contained in that loadset are displayed in the following example. Notice that the QAE141 program is followed by EXC in the example. These characters indicate that the program was removed from the loadset using the ZOLDR EXCLUDE command.

```
User:   ZOLDR DISP L CAPRES

System: OLDR1200I 17.25.41 CLE2 - BEGIN LOADSET INFORMATION
         LOADSET   STATUS
         CAPRES    CREATED ON 04/28/94 08.17.18
                   ACTIVATED ON CPUID C ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID D ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID E ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID Z ON 04/28/94 08.26.08
                   ACTIVATED ON CPUID 0 ON 04/28/94 08.26.08

                   CONTAINS THESE PROGRAMS
                   QEA041      QEA141-EXC
        CLE2 - END OF DISPLAY
```

A list of the loadsets that intersect with the FIX1018 loadset is displayed in the following example.

```
User:   ZOLDR DISP INTER FIX1018

System: OLDR3404I 19.39.59 CLEO - LOADSET FIX1018 INTERSECTS WITH THESE PREVIOUSLY
                 ACTIVATED LOADSETS-
                 LOADSET     PRORAM-INTERSECTIONS
                 FIX1017     CVG1GA   UOL8GA

        END OF DISPLAY
```

The rules used during E-type loader processes are displayed in the following example.

```
User:   ZOLDR DISP RULES

System: OLDR5300I 19.39.59 CLDV - E-TYPE LOADER RULES
                   PROGCHAR  FR SHARED 31BIT KEY0 RESTRICT MONTC CMB
        END OF DISPLAY
```

The values defined for the variables used during E-type loader processes are displayed in the following example. Note that the number of extra program allocation table (PAT) slots shown is for the current IPL.

```
User:   ZOLDR DISP VALUES

System: OLDR5101I 19.39.59 CLDU - E-TYPE LOADER VALUES
                    NUMBER OF EXTRA PAT SLOTS ALLOCATED        -       1000
                    NUMBER OF EXTRA PAT SLOTS AVAILABLE        -       1000
                         EXTRA PAT THRESHOLD PERCENT           -         10
                    NUMBER OF FIXED FILE RECORDS               -       2000
                         FIXED FILE RECORD THRESHOLD PERCENT   -         10
                    NUMBER OF INCOMPATIBLE PAT SLOTS TO REPORT -         10
                    POLICE ROUTINE TIME INTERVAL, IN SECONDS   -          3
                    JOB DETECTION TIME INTERVAL, IN SECONDS    -          6
        END OF DISPLAY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR EXCLUDE–Exclude Programs from a Loadset

Use this command to remove a program from a loadset. When you remove a program from a loadset, E-type loader functions no longer affect that program until you include the program in the loadset again using the E-type loader reinclude function.

Requirements and Restrictions

None.

Format

```
                               ┌─.────────┐
                               ▼          │
►►──ZOLDR Exclude── ──lsname── ───progname┴───────────────────────────►◄
```

lsname
is the 5- to 8-character alphanumeric name of the loadset that contains the program you want to remove.

progname
is the 4-character alphanumeric name of the program you want to remove from the loadset. You can use an asterisk (*) as a wildcard character for this parameter.

Notes:

1. You can specify as many as 20 program names at one time.
2. You can remove no more than 4000 programs from a loadset at one time. Therefore, use care if you specify an * for a wildcard character because you may exceed this limit.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- You can remove programs from both active and inactive loadsets.
- If you remove a program from an active loadset, the program is first deactivated on each processor where the loadset is activated and then it is removed from the loadset.
- If you remove programs from a loadset and then delete that loadset, the programs you removed are deleted as well.

Examples

The RSD4 program is removed from the TEST1 loadset in the following example. The TEST1 loadset is active on all processors in the TPF system.

```
User:   ZOLDR EXCLUDE TEST1 RSD4

System: OLDR2041I 17.25.41 EXCLUDE FROM LOADSET TEST1 SCHEDULED
         FOR CPU B
        OLDR2041I 17.25.41 EXCLUDE FROM LOADSET TEST1 SCHEDULED
         FOR CPU C
        OLDR5812I 17.25.41 COLJ - EXCLUDE COMPLETED
        THE FOLLOWING PROGRAMS ARE EXCLUDED FROM LOADSET TEST1
          RSD4
        ANY DUPLICATE NAMES IGNORED
        END OF DISPLAY
        OLDR2048I 17.25.41 EXCLUDE FROM LOADSET TEST1 COMPLETED
         FOR CPU B  AS REQUESTED BY CPU B
        OLDR2048I 17.25.41 EXCLUDE FROM LOADSET TEST1 COMPLETED
         FOR CPU C  AS REQUESTED BY CPU B
```

The BXAA program is removed from the PARLOT loadset in the following example. The PARLOT loadset is not active on any processors in the TPF system.

```
User:   ZOLDR EXCLUDE PARLOT BXAA

System: OLDR5812I 19.39.59 COLJ - EXCLUDE COMPLETED
        THE FOLLOWING PROGRAMS ARE EXCLUDED FROM LOADSET PARLOT
          BXAA
        ANY DUPLICATE NAMES IGNORED
        END OF DISPLAY
```

All programs beginning with the letter C are removed from the LOAD2A loadset in the following example.

```
User:   ZOLDR EXCLUDE LOAD2A C*

System: OLDR5812I 19.39.59 COLJ - EXCLUDE COMPLETED
        THE FOLLOWING PROGRAMS ARE EXCLUDED FROM LOADSET LOAD2A
          CERA  CERC  CVZZ
        ANY DUPLICATE NAMES IGNORED
        END OF DISPLAY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR LOAD–Load a Loadset

Use this command to load loadsets, or changes to the program allocation table (PAT), to the online system. You can load loadsets and PAT changes from a tape, general data set, virtual reader, or other user-defined input device.

Requirements and Restrictions

- Before you can enter this command, you must assign a data definition name to the input device using the ZDSMG DEFINE or ZDSMG MT command.
- If you are loading loadsets from a tape input device, you must mount the tape input device using the ZTMNT command.
- If you are loading loadsets from a virtual reader input device, you can load only one file from the virtual reader at a time. Therefore, you cannot load loadsets from an input device that was defined as VRDR-ALL.

Format

```
►►──ZOLDR Load── ──ddname──┬────────────────┬──────────────────────────►◄
                           │    ┌─.───────┐ │
                           │    ▼         │ │
                           ├─ ───lsname───┴─┤
                           └─ ──Patu────────┘
```

ddname
: is the 1- to 16-character alphanumeric data definition name of the input device (tape, general data set, virtual reader, or user-defined device) that contains the loadsets you want to load.

lsname
: is the 5- to 8-character alphanumeric name of the loadset that you want to load. If you do not specify a loadset name, all of the loadsets on the input device are loaded.

Note: You can specify as many as 20 loadset names at one time.

Patu
: allocates new programs to the online system by loading changes to the PAT from the input device. Loadsets contained on the input device are not loaded.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOLDR HELP
  ZOLDR ?
- Once a loadset is loaded, you must activate it using the ZOLDR ACTIVATE command in order to make the programs in the loadset available to the processors in the TPF system.

Examples

All the loadsets on the TESTDSN input device are loaded to the TPF system in the following example.

```
User:   ZOLDR LOAD TESTDSN

System: OLDR1016I 19.39.59 CELA - LOAD REQUEST RECEIVED
        OLDR3000I 19.39.59 CLD0 - LOADSET FIX1018 HAS BEEN LOADED FROM DDNAME TESTDSN
        OLDR3000I 19.39.59 CLD0 - LOADSET LLTPKG HAS BEEN LOADED FROM DDNAME TESTDSN
        OLDR3001I 19.39.59 CELL - LOAD FOR DDNAME TESTDSN COMPLETED
```

The TOTAL, CAPRES, and LUAPPL loadsets on the VRDR1 input device are loaded in the following example.

```
User:   ZOLDR LOAD VRDR1 TOTAL.CAPRES.LUAPPL

System: OLDR1016I 19.39.59 CELA - LOAD REQUEST RECEIVED
        OLDR3000I 19.39.59 CLD0 - LOADSET TOTAL HAS BEEN LOADED FROM DDNAME VRDR1
        OLDR3000I 19.39.59 CLD0 - LOADSET CAPRES HAS BEEN LOADED FROM DDNAME VRDR1
        OLDR3000I 19.39.59 CLD0 - LOADSET LUAPPL HAS BEEN LOADED FROM DDNAME VRDR1
        OLDR3001I 19.39.59 CELL - LOAD FOR DDNAME VRDR1 COMPLETED
```

The CAPRES loadset was already loaded in the following example.

```
User:   ZOLDR LOAD VRDR1 TOTAL.CAPRES.LUAPPL

System: OLDR1016I 19.39.59 CELA - LOAD REQUEST RECEIVED
        OLDR3000I 19.39.59 CLD0 - LOADSET TOTAL HAS BEEN LOADED FROM DDNAME VRDR1
        OLDR3008W 19.39.59 CLD0 - LOADSET CAPRES SKIPPED FOR DDNAME VRDR1 - LOADSET
                  ALREADY EXISTS ON ONLINE SYSTEM
        OLDR3000I 19.39.59 CLD0 - LOADSET LUAPPL HAS BEEN LOADED FROM DDNAME VRDR1
        OLDR3001I 19.39.59 CELL - LOAD FOR DDNAME VRDR1 COMPLETED
```

In the following example, new programs are allocated by loading changes to the PAT.

```
User:   ZOLDR LOAD NEWPGMS PATU

System: OLDR1016I 19.39.59 CELA - LOAD REQUEST RECEIVED
        OLDR3001I 19.39.59 CELL - LOAD FOR DDNAME NEWPGMS COMPLETED
        OLDR4100I 19.39.59 CLEX - NEW PROGRAMS REPLACED SPARE ENTRIES IN
                  CORE PAT - SEE RO FOR DETAILS
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR RECLAIM–Build and Recover E-Type Loader Records

Use this command to build the database containing the E-type loader fixed file records and to recover any E-type loader fixed file records that are lost during an initial program load (IPL).

For example, use the ZOLDR RECLAIM command when you load a new file address compute routine (FACE) table that has additional E-type loader fixed file records defined. You can also use the ZOLDR RECLAIM command if there is a decrease in the number of E-type loader fixed file records available, but this decrease is not associated with an increased number of loadsets that are loaded to the TPF system.

Requirements and Restrictions

You cannot enter this command while the E-type loader load function is running on the subsystem.

Format

```
►►──ZOLDR RECLaim──────────────────────────────────────────────────────────►◄
```

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZOLDR HELP
ZOLDR ?

Examples

The E-type loader database is rebuilt in the following example.

```
User:   ZOLDR RECLAIM

System: OLDR1016I 19.39.59 CELA - RECLAIM REQUEST RECEIVED
        OLDR0206I 19.39.59 COL1 - E-TYPE LOADER RECLAIM STARTED ON CPU-B
        OLDR0213I 19.39.59 COL0 - E-TYPE LOADER RECLAIM COMPLETED ON CPU-B,
                 1802 OUT OF     2000 OLD1 RECORDS AVAILABLE
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOLDR REINCLUDE–Reinclude Programs in a Loadset

Use this command to include programs in a loadset if they were previously removed from the loadset using the E-type loader exclude function.

Requirements and Restrictions

- You can include programs in a loadset only if the programs were previously removed from the loadset using the ZOLDR EXCLUDE command.
- You must deactivate the loadset on all processors before you can include programs in it.

Format

```
                                    ┌─.────────┐
                                    ▼          │
►►──ZOLDR REInclude── ──lsname── ───progname───┴──────────────────────────►◄
```

lsname
is the 5- to 8-character alphanumeric name of the loadset where you want to include the program.

progname
is the 4-character alphanumeric name of the program you want to include in the loadset. You can use an asterisk (*) as a wildcard character for this parameter.

Notes:

1. You can specify as many as 20 program names at one time.
2. You can include no more than 4000 programs in a loadset at one time. Therefore, use care if you specify an * for a wildcard character because you may exceed this limit.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZOLDR HELP
ZOLDR ?

Examples

In the following example, the CLDX program is included in the SECOND loadset again. Note that the CLDX program was previously removed from the loadset using the E-type loader exclude function.

```
User:   ZOLDR REINCLUDE SECOND CLDX

System: OLDR5813I 19.39.59 COLG - REINCLUDE COMPLETED
        THE FOLLOWING PROGRAMS RE-INCLUDED IN LOADSET SECOND
           CLDX
        ANY DUPLICATE NAMES IGNORED
        END OF DISPLAY
```

ZOLDR REINCLUDE

In the following example, the CLDX, COLN, and CEL2 programs are included in the SECOND loadset again. Note that all 3 programs were previously removed from the loadset using the E-type loader exclude function.

```
User:   ZOLDR REINCLUDE SECOND CLDX.COLN.CEL2

System: OLDR5813I 19.39.59 COLG - REINCLUDE COMPLETED
        THE FOLLOWING PROGRAMS RE-INCLUDED IN LOADSET SECOND
           CLDX  COLN  CEL2
        ANY DUPLICATE NAMES IGNORED
        END OF DISPLAY
```

Related Information

See *TPF System Installation Support Reference* for more information about the E-type loader.

ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition

Use this command to change the attributes of a data store (DS) or to change the attributes of a data definition (DD) for a specific data store.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) and define the target data store before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZOODB CHAnge── ──┬─DS── ──dsname──┤ DS Options ├──────────────────────┬──►◄
                     └─DD── ──ddname── ──DS-dsname──┤ DD Options ├─┘

DS Options:

├──┬─────────────────┬──┬─────────────────────────┬──────────────────────────┤
   ├─ ──INVENTORY────┤  └─ ──DELETE-──┬─DELAY──┤
   └─ ──NOINVENTORY──┘                └─IMMED──┘

DD Options:

                        ┌─────────────────────────┐
                        ▼                         │
├──┬──────────────┬──┬─────────────────────────┬──┬──────────────┬─────────┤
   ├─ ──NOSHADOW──┤  ├─ ──DATARID-recordid─────┤  ├─ ──FORCE─────┤
   └─ ──SHADOW────┘  ├─ ──DIRECTRID-recordid───┤  └─ ──NOFORCE───┘
                     └─ ──INDEXRID-recordid────┘
```

DS
: changes a data store.

dsname
: is the 1- to 8-character name of the data store.

DD
: changes a data definition.

ddname
: is the 1- to 32-character name of the data definition.

INVENTORY
: specifies that a collection inventory will be maintained for this data store. If no collection inventory exists, it will be created.

Note: Only persistent identifiers (PIDs) for collections that are created after the inventory has been created will be added to the inventory.

NOINVENTORY
specifies that no collection inventory will be used for this data store. If there is a collection inventory, it will be deleted.

DELETE
determines the disposition of deleted collections, where:

DELAY
specifies that a deleted collection will be added to the DS_DELETED data store system collection and will not be deleted from the system for 48 hours. The collection can be reclaimed during the 48-hour interval. All pending deletes can be processed by entering ZBROW COL EMPTY DS_DELETED.

IMMED
specifies that a deleted collection will be immediately returned to the system and cannot be reclaimed. If there are deletions pending, they are processed immediately.

NOSHADOW
specifies that no shadowing will be used for this collection.

SHADOW
specifies the use of shadowing for this collection.

DATARID
defines the record ID for the data records.

DIRECTRID
defines the record ID for the directory records.

INDEXRID
defines the record ID for the index records.

recordid
is a 4-character hexadecimal record ID.

FORCE
builds the collection using an extended structure rather than the normal compact structure.

NOFORCE
builds the collection with the normal compact structure.

Additional Information

None.

Examples

The BATCHWRK data store (DS) was created with default options that caused it to be created with an inventory and delayed deletions active. It was determined that the inventory or the delayed deletions are not wanted and, therefore, should be removed. In the following example, the data store is changed to remove both functions.

```
User:   ZOODB CHA DS BATCHWRK NOINVENTORY DELETE-IMMED

System: OODB0014I 12.15.29 DATA STORE BATCHWRK CHANGED
        OODB0030I 12.15.29 CHANGE COMPLETED FOR DS BATCHWRK
```

The record IDs for data and directory records in the CUSTOMER_INDEX data definition are changed in the following example.

```
User:   ZOODB CHA DD CUSTOMER_INDEX DS-BANK1_DS DATARID-FC10 DIRECTRID-FC15

System: OODB0032I 12.15.29 DD CUSTOMER_INDEX CHANGED
        OODB0031I 12.15.29 CHANGE COMPLETED FOR DD CUSTOMER_INDEX
```

Related Information

- See “ZBROW COLLECTION–Perform Maintenance on a Collection” on page 123 for more information about the ZBROW COLLECTION command.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOODB DEFINE–Define Data Store or Data Definition

Use this command to define a unique data store (DS) or to define the attributes of a new data definition (DD) for a specific data store.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) before you enter this command.
- A data store definition is systemwide. Data stores are ***not*** subsystem unique. Therefore, the data store name ***must*** be unique across the TPF system complex if you specify the DS parameter.
- If you specify the DD parameter, the target data store ***must*** be defined.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZOODB DEFine── ──┬─DS── ──dsname──┤ DS Options ├────────────────┬──────────►◄
                     └─DD── ──ddname── ──DS-dsname──┤ DD Options ├──┘
```

DS Options:

```
   ┌─ ──INVENTORY───┐  ┌─ ──DELETE-DELAY─┐
├──┼────────────────┼──┼─────────────────┼──────────────────────────────────────┤
   └─ ──NOINVENTORY─┘  └─ ──DELETE-IMMED─┘
```

DD Options:

```
                      ┌──────────────────────────────┐
   ┌─ ──NOSHADOW─┐    ▼                              │  ┌─ ──NOFORCE─┐
├──┼─────────────┼────────┬───────────────────────┬──┴──┼────────────┼──────────┤
   ├─ ──SHADOW───┤        ├─ ──DATARID-recordid───┤     └─ ──FORCE───┘
   └─ ──TEMP─────┘        ├─ ──DIRECTRID-recordid─┤
                          └─ ──INDEXRID-recordid──┘
```

DS
: defines a data store.

dsname
: is the 1- to 8-character name of the data store.

DD
: defines a data definition.

ddname
: is the 1- to 32-character name of the data definition.

INVENTORY
: specifies that a collection inventory will be maintained for this data store.

TPFCS will add the persistent identifier (PID) of every collection created for this data store to this inventory when the collection is created.

NOINVENTORY
specifies that no collection inventory will be maintained for this data store.

DELETE
determines the disposition of deleted collections, where:

DELAY
specifies that a deleted collection will be added to a data store collection and will not be deleted from the system for 48 hours. The collection can be reclaimed during the 48-hour interval.

IMMED
specifies that a deleted collection will be immediately returned to the system. The collection cannot be reclaimed.

NOSHADOW
specifies that shadowing will not be used for this collection.

SHADOW
specifies the use of shadowing for this collection.

TEMP
defines a temporary collection. Any record IDs specified must be for a short-term pool file.

DATARID
defines the record ID for the data records.

Note: The ID ***must*** be defined in the record ID attribute table (RIAT).

recordid
is a 4-character hexadecimal record ID.

DIRECTRID
defines the record ID for the internal directory record.

Note: The ID ***must*** be defined in the RIAT.

INDEXRID
defines the record ID for the internal index record.

Note: The ID ***must*** be defined in the RIAT.

FORCE
forces TPF collection support to start building the collection using an extended structure rather than the normal compact structure.

NOFORCE
builds the collection with the normal compact structure.

Additional Information

None.

Examples

The BANK1_DS data store (DS) is defined in the following example to have an inventory and delayed deletes.

```
User:   ZOODB DEF DS BANK1_DS

System: OODB0012I 11.55.40 DATA STORE BANK1_DS DEFINED
        OODB0011I 11.55.40 REQUEST COMPLETED FOR DS BANK1_DS
```

The BATCHWRK data store is defined in the following example to have no inventory and delayed deletes.

```
User:   ZOODB DEF DS BATCHWRK NOINVENTORY DELETE-IMMED

System: OODB0012I 11.55.40 DATA STORE BATCHWRK DEFINED
        OODB0011I 11.55.40 REQUEST COMPLETED FOR DS BATCHWRK
```

The CUSTOMER_INDEX data definition is defined in the following example.

```
User:   ZOODB DEF DD CUSTOMER_INDEX DS-BANK1_DS DATARID-FC00 DIRECTRID-FC01

System: OODB0007I 08.25.23 DD CUSTOMER_INDEX DEFINED
        OODB0006I 08.25.23 REQUEST COMPLETED FOR DD CUSTOMER_INDEX
```

Related Information

- See “ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition” on page 1083 for more information about the ZOODB CHANGE command.
- See “ZOODB DELETE–Delete Data Store or Data Definition” on page 1089 for more information about the ZOODB DELETE command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  – *TPF Application Programming*
  – *TPF Concepts and Structures*
  – *TPF Database Reference*.

ZOODB DELETE–Delete Data Store or Data Definition

Use this command to delete a specific data store or to delete a data definition from a specific data store.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZOODB DELete── ──┬DS── ──dsname──────────────────┬──────────────────────────►◄
                     └DD── ──ddname── ──DS-dsname────┘
```

DS
: deletes a data store.

dsname
: is the 1- to 8-character name of the data store.

DD
: deletes a data definition.

ddname
: is the 1- to 32-character name of the data definition to be deleted.

Additional Information

- Deleting a data store deletes the data store entry from the TPF system dictionary and deletes all system collections and data definitions in that data store. All resources used by these collections are returned to the system and none of these collections can be reclaimed.
- You do ***not*** have to enter this command from the subsystem that created the data store.
- Deleting a data store does not delete user collections. You must delete all user collections before entering this command with the DS parameter specified or the pool records used by these collections will be lost until the next recoup is run.
- You cannot delete the TPFDB data store.
- Deleting a data definition has no effect on any collections that were created by using the deleted data definition.

Examples

The CUSTOMER data store is deleted in the following example.

```
User:   ZOODB DEL DS CUSTOMER

System: OODB0047I 11.15.33 DELETE OF DS CUSTOMER COMPLETED
```

The CUSTOMER_INDEX data definition is deleted in the following example.

```
User:   ZOODB DEL DD CUSTOMER_INDEX DS-BANK1_DS

System: OODB0042I 12.24.36 DD CUSTOMER_INDEX DELETED
        OODB0041I 12.24.36 DELETE COMPLETED FOR DD CUSTOMER_INDEX
```

Related Information

- See “ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition” on page 1083 for more information about the ZOODB CHANGE command.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOODB DISPLAY–Display the Attributes of a Data Definition or Data Store

Use this command to display the attributes of a data definition (DD) or data store (DS), or the names of all the defined data definitions for a specific data store.

Requirements and Restrictions

You ***must*** initialize TPF collection support (TPFCS) and define the target data store before you enter this command.

Format

```
►►──ZOODB DISplay── ──┬─DS── ──┬─dsname─┬──────────────────────────┬────────────►◄
                      │        └─ALL────┘                          │
                      └─DD── ──┬─ddname─┬── ──DSname-dsname─┘
                               └─ALL────┘
```

DS
: displays one or more data stores.

dsname
: is the 1- to 8-character name of the data store.

DD
: displays one or more data definitions.

ddname
: is the 1- to 32-character name of the data definition to be deleted.

ALL
: displays the names of all defined data stores or all defined data definitions for the data store.

DSname
: specifies the target data store.

Additional Information

None.

Examples

The BANK_DS data store definition is shown in the following example. The following fields describe the data store attributes of interest:

OBJECT_SEQ_CTR
: indicates the number of times the data store object that contains the data store information has been updated. This number includes any changes made to the data store object when the TPF system created it.

DB_IDSNAME
: is the name of the data store to which the data definition belongs.

DB_IDICT_OID
: contains the PID of the data store user dictionary (DS_USER_DICT).

DB_IDICT_SYSTEM
contains the PID of the system dictionary (DS_SYSTEM_DICT) for the data store.

DB_OID_INVENTORYOID
contains the PID of the inventory collection (DS_INVENTORY) for the data store, or 0 if the data store does not have an inventory.

DB_DELETED_PID
contains the PID of the deleted PID collection (DS_DELETED) for the data store.

DB_IDICT_RECOUP
contains the PID of the DS_RECOUP collection for the data store. This collection contains information about the recoup indexes created in the data store.

DB_IDICT_BROWSER
contains the PID of the browser keysortedset for the data store. This collection contains the names and corresponding PIDs of all named collections.

DB_I_RESTARTLOG_PID
contains the PID of the restart log for the data store, or 0 if there is no restart log for the data store.

DB_DELETE_FLAG
is one of the following:

00 indicates that the data store uses delayed deletion.

01 indicates that the data store uses immediate deletion.

Note: All fields displayed by this command are internal to TPFCS and are subject to change.

```
User:   ZOODB DISP DS BANK_DS

System: OODB0025I 10.33.47 START OF ATTRIBUTE DISPLAY
        PART  NAME  **********            DB_OBJ
        CLASS NAME  **                    OBJECT
        OBJECT_ID                         00000028
        OBJECT_LGH                        00000168
        OBJECT_SEQ_CTR                    00000001
        CLASS NAME  **                    ObjectPart
        OBJ_Part_CHANGE                   00
        OBJ_Part_RELEASE                  00
        OBJ_Part_RESERVE2                 0000
        OBJ_Part_PartID                   00000000
        OBJ_Part_OIE                      00000000
        OBJ_Part_Faddr                    00
        CLASS NAME  **                    DB_OBJ
        DB_CHAIN_Next                     01001C58
        DB_CHAIN_Prev                     01001C58
        DB_USER_COUNT                     00000001
        DB_IDSNAME                        BANK_DS
        DB_IDICT_OID                      0202FC16 B696D9C0 C2C1D5D2 6DC4E240
        ****                              08653C4D 08653C51 00000000 00000000
        DB_IDICT_SYSTEM                   0202FC16 B696D9C0 C2C1D5D2 6DC4E240 _
        ****                              08653C55 08653C59 00000000 00000000
        DB_OID_INVENTORYOID               0202FC16 B696D9C0 C2C1D5D2 6DC4E240
        ****                              08653C45 08653C49 00000000 00000000
        DB_DELETED_PID                    0202FC16 B696D9C0 C2C1D5D2 6DC4E240 _
        ****                              08653C5D 08653C61 00000000 00000000
        DB_IDICT_RECOUP                   0202FC16 B696D9C0 C2C1D5D2 6DC4E240
        ****                              08653C65 08653C69 00000000 00000000
        DB_IDICT_BROWSER                  0202FC16 B696D9C1 C2C1D5D2 6DC4E240 _
        ****                              08653C6D 08653C71 00000000 00000000
        DB_I_RESTARTLOG_PID               0202FC16 B696D9C1 C2C1D5D2 6DC4E240
        ****                              08653C75 08653C79 00000000 00000000
        DB_SSNAME                         BSS
        DB_SSINDEX                        FF00
        DB_SSUINDEX                       FF00
        DB_IOID_FORMAT                    02
        DB_TRACE_FLAG                     Y
        DB_DELETE_FLAG                    00
        DB_DELOID_DELETE_DLAY                        0
        END OF DISPLAY
        OODB0022I 10.34.00 DISPLAY COMPLETED FOR DS BANK_DS +
```

A display of all data store names, the subsystem in which they reside, and the persistent identifier (PID) of their inventory collection is shown in the following example.

```
User:   ZOODB DISP DS ALL

System: OODB0021I 11.02.26 DATA STORE LIST DISPLAY
         DS - TPFDB       SS - BSS
         INVENTORY - 0002FC16 AF724045 E3D7C6C4 C2404040
                     18034AD8 18034AD9 00000000 00000000

         DS - BANK1_DS    SS - BSS
         INVENTORY - 0202FC16 AF724051 C2C1D5D2 F16DC4E2
                     18034B03 18034B04 00000000 00000000

         DS - BANK2_DS    SS - WP
         INVENTORY - 0202FC16 AF7B94DA C2C1D5D2 F24BC4E2
                     00000000 18039398 00000000 18039399

        END OF DISPLAY
        OODB0022I 11.02.26 DISPLAY COMPLETED FOR DS
```

The example that follows shows the data definition CUSTOMER_INDEX. The following fields, which are shown in the display, can be set externally:

DDEF_DNAME
is the name of the data definition.

DDEF_DSNAME
is the name of the data store to which this data definition belongs.

DDEF_TEMPC
is one of the following:

Y indicates that the data definition is for temporary collections.

N indicates that the data definition is for persistent collections.

DDEF_RECID_DATA
is the record ID for data records.

DDEF_RECID_INDEX
is the record ID for index records.

DDEF_RECID_DIRECT
is the record ID for directory records.

DDEF_SHADOW
is one of the following:

00 indicates that collections using this data definition are not shadowed.

01 indicates that collections using this data definition are shadowed.

DDEF_FORCE
is one of the following:

00 indicates that collections using this data definition are left as compact structures.

01 indicates that collections using this data definition are forced into extended structures.

Note: All fields displayed by this command are internal to TPFCS and are subject to change.

```
User:   ZOODB DISP DD CUSTOMER_INDEX DSN-BANK1_DS

System: OODB0025I 12.49.35 START OF ATTRIBUTE DISPLAY
        PART NAME                           DB_OBJ
        CLASS NAME                          OBJECT
        OBJECT_ID                           00000270
        OBJECT_LGH                          000000B0
        OBJECT_SEQ_CTR                      00000000
        CLASS NAME                          ObjectPart
        OBJ_Part_CHANGE                     00
        OBJ_Part_RESERVE2                   000000
        OBJ_Part_PartID                     00000000
        OBJ_Part_OIE                        00000000
        DDEF_DNAME                          CUSTOMER_INDEX
        *****
        DDEF_DSNAME                         BANK1_DS
        DDEF_DASDC                          Y
        DDEF_TEMPC                          N
        DDEF_PRTC                           N
        DDEF_POOLC                          Y
        DDEF_STRUCT_NAME
        *****
        DDEF_TPFDF_NAME
        DDEF_RECID_DATA                     FC10
        DDEF_RECID_INDEX                    0000
        DDEF_RECID_DIRECT                   FC15
        DDEF_POOL_TYPE                      L
        DDEF_SHADOW                         00
        DDEF_FORCE                          00

        OODB0027I 12.04.32 DISPLAY COMPLETED FOR DD CUSTOMER_INDEX
```

The following example shows all the data definition names for data store BANK1_DS.

```
User:   ZOODB DISP DD ALL DS-BANK1_DS

System: OODB0026I 11.18.32 DATA DEFINITION LIST DISPLAY
         DD - CUSTOMER_INDEX
         DD - DEFAULT_ARRAY
         DD - DEFAULT_BAG
         DD - DEFAULT_BYTE
         DD - DEFAULT_DICTIONARY
         DD - DEFAULT_KEYBAG
         DD - DEFAULT_KEYEDLOG
         DD - DEFAULT_KEYSET
         DD - DEFAULT_LOG
         DD - DEFAULT_ORDERED
         DD - DEFAULT_PROPERTY
         DD - DEFAULT_SET
         DD - DEFAULT_SORTED
         DD - DEFAULT_TEMP_ARRAY
         DD - DEFAULT_TEMP_BAG
         DD - DEFAULT_TEMP_BYTE
         DD - DEFAULT_TEMP_DICTIONARY
         DD - DEFAULT_TEMP_KEYBAG
         DD - DEFAULT_TEMP_KEYEDLOG
         DD - DEFAULT_TEMP_KEYSET
         DD - DEFAULT_TEMP_LOG
         DD - DEFAULT_TEMP_ORDERED
         DD - DEFAULT_TEMP_SET
         DD - DEFAULT_TEMP_SORTED
         DD - PRT_OUTPUT
        OODB0027I 12.19.07 DISPLAY COMPLETED FOR DD
```

Related Information

- See "ZOODB INIT–Initialize TPF Collection Support" on page 1097 for more information about initializing TPFCS.

- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOODB INIT–Initialize TPF Collection Support

Use this command to initialize TPF collection support (TPFCS) for the first time.

Requirements and Restrictions

- You can enter this command only once.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
►►──ZOODB INIT───────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example shows how to initialize TPF collection support.

```
User:   ZOODB INIT

System: OODB0001I 14.43.55 TO2 INITIALIZED WITHOUT ERRORS
        OODB0002I 14.43.55 TO2 INITIALIZATION REQUEST COMPLETED
```

Related Information

- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOODB MIGRATE–Migrate a Data Store

Use this command to migrate a specific data store to the current TPF collection support (TPFCS) collection format with new pool addresses assigned to the collections.

Requirements and Restrictions

- You ***must*** initialize TPFCS before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.
- You ***must*** enter this command from the subsystem that owns the data store to be migrated.

Format

```
►►──ZOODB MIGrate── ──DS── ──dsname── ──────────────────────────────────────►◄
```

DS
migrates a data store.

dsname
is the 1- to 8-character name of the data store.

Additional Information

- This command does the following:
  - Re-creates all system collections in a data store, including:
    - Any collections in the #INODE fixed file records when a file system data store (IFSX....) is migrated
    - Any MATIP definitions when the MATIP_DS data store is migrated.
  - Copies the contents of the old system collections to new collections one element at a time
  - Deletes the old system collections.
- User collections are not migrated with this command. Migrate all user collections for a data store before entering this command.
- To migrate, do the following:
  1. Make sure you have enough pools.
  2. Enter **ZOODB DISP DS ALL** to get a list of all defined data stores.
  3. Migrate or copy all user collections in each of the target data stores.
  4. Enter the ZOODB MIGRATE command for each user data store from the subsystem that owns that data store. Migrate the TPFDB data store last.
  5. If you are migrating a TPF recoup data store (IRCP....), you ***must*** enter the **ZRECP SETUP** command to complete the migration.
  6. If you are migrating a TPF MQSeries data store (MQSC ....), you must also enter a ZMQSC DBREBUILD command to migrate the specfic MQSeries system collections within that data store.

Examples

The BANK1_DS data store is migrated in the following example.

```
User:   ZOODB MIGRATE DS BANK1_DS

System: OODB0062I 11.15.33 MIGRATE OF DS BANK1_DS COMPLETED
```

Related Information

- See "ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition" on page 1083 for more information about the ZOODB CHANGE command.
- See "ZOODB DEFINE–Define Data Store or Data Definition" on page 1086 for more information about defining a target data store.
- See "ZOODB DELETE–Delete Data Store or Data Definition" on page 1089 for more information about the ZOODB DELETE command.
- See "ZOODB INIT–Initialize TPF Collection Support" on page 1097 for more information about initializing TPFCS.
- See "ZOODB RECREATE–Re-Create a Data Store" on page 1100 for more information about the ZOODB RECREATE command.
- See "ZMQSC DBREBUILD–Rebuild TPF MQSeries Definitions" on page 857 for more information about the ZMQSC DBREBUILD command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOODB RECREATE–Re-Create a Data Store

Use this command to re-create a specific data store to its initialized state.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) pools enabled before you can use this command. Pools are not enabled until the system is cycled to CRAS state or above.
- You ***must*** enter this command from the subsystem that owns the data store to be re-created.

Format

```
►►──ZOODB RECreate── ──DS── ──dsname── ──────────────────────────────►◄
```

DS
re-creates a data store.

dsname
is the 1- to 8-character name of the data store.

Additional Information

- This command does the following:
  - Re-creates and replaces all system collections contained in the data store using new pool file addresses
  - Deletes old system collections, except for the old recoup indexes. The pools used by the old recoup index collections will be lost until the next recoup is run.
- All user collections and recoup indexes will be lost.
- If you do not delete all user collections before entering this command, the pool records associated with those collections will not be available for use until the next recoup is run.
- You cannot re-create the TPFDB data store.

Examples

The TEST1_DS data store is re-created in the following example.

```
User:   ZOODB RECREATE DS TEST1_DS

System: OODB0072I 11.15.33 RE-CREATE OF DS TEST1_DS COMPLETED
```

Related Information

- See “ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition” on page 1083 for more information about the ZOODB CHANGE command.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.

- See “ZOODB DELETE–Delete Data Store or Data Definition” on page 1089 for more information about the ZOODB DELETE command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See the following books for more information about TPFCS:
  – *TPF Application Programming*
  – *TPF Concepts and Structures*
  – *TPF Database Reference*.

ZOODB SET–Set TPF Collection Support

Use this command to set the method trace table or the start of a dump on a `TO2_getErrorText` function call to on or off.

Requirements and Restrictions

- You ***must*** initialize TPF collection support (TPFCS) before you enter this command.
- The TPF system ***must*** have get file storage (GFS) enabled before you can use this message. Pools are not enabled until the system is cycled to CRAS state or above.

Format

```
                              ┌─  ──────────────────────┐
                              ▼                         │
►►──ZOODB SET── ──TO2── ──────┬─MTRace-──┬─ON──┬─────┬──┴──────────────────────►◄
                              │          └─OFF─┘     │
                              └─TEXTdump-──┬─ON──┬───┘
                                           └─OFF─┘
```

TO2
: sets TPFCS method trace table or dump creation.

MTRace
: specifies one of the following:

  ON
  : activates the method trace table.

  OFF
  : deactivates the method trace table.

TEXTdump
: specifies one of the following:

  ON
  : activates dump creation on a `TO2_getErrorText` function call.

  OFF
  : deactivates dump creation on a `TO2_getErrorText` function call.

Additional Information

- To display the current settings for the MTRACE and TEXTDUMP parameters, enter **ZBROW CLASS ATTR NAME-TO2**.
- If you do not specify this command, the method trace table is active and dump creation for a `TO2_getErrorText` function call is not active.

Examples

The following example shows how to deactivate the method trace table and the start of a dump.

```
User:   ZOODB SET TO2 MTRACE-OFF TEXTDUMP-OFF

System: OODB0051I 18.23.20 SET REQUEST COMPLETED
```

Related Information

- See “ZOODB CHANGE–Change the Attributes of a Data Store or Data Definition” on page 1083 for more information about the ZOODB CHANGE command.
- See “ZOODB DEFINE–Define Data Store or Data Definition” on page 1086 for more information about defining a target data store.
- See “ZOODB DELETE–Delete Data Store or Data Definition” on page 1089 for more information about the ZOODB DELETE command.
- See “ZOODB INIT–Initialize TPF Collection Support” on page 1097 for more information about initializing TPFCS.
- See “ZOODB MIGRATE–Migrate a Data Store” on page 1098 for more information about the ZOODB MIGRATE command.
- See “ZOODB RECREATE–Re-Create a Data Store” on page 1100 for more information about the ZOODB RECREATE command.
- See the following books for more information about TPFCS:
  - *TPF Application Programming*
  - *TPF Concepts and Structures*
  - *TPF Database Reference*.

ZOSAE–Manage an OSA-Express Connection

Use this command to do the following:

- Define the definitions for Open Systems Adapter (OSA)-Express connections
- Modify the definitions for an OSA-Express connection
- Delete the definition for an OSA-Express connection
- Define a virtual IP address (VIPA) and associate it with a specific OSA-Express connection
- Delete a VIPA definition and disassociate it from the OSA-Express connection
- Display the definition and status of an OSA-Express connection
- Display the number of packets sent and received across each OSA-Express connection in a 5-second interval
- Swing all the VIPAs associated with a primary OSA-Express connection to its alternate OSA-Express connection, or from the alternate OSA-Express connection to its primary OSA-Express connection.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).
- To define an OSA-Express connection, you must first enter the ZOSAE command with the DEFINE parameter specified with an OSA-Express connection name that has not been defined previously.
- A primary OSA-Express connection and its alternate OSA-Express connection cannot be connected to the same OSA-Express card if you are using virtual IP addresses (VIPAs).
- To eliminate single points of failure, connect primary OSA-Express connections and alternate OSA-Express connections through different OSA-Express cards to different Ethernets.

Format

```
►►──ZOSAE── ──────────────────────────────────────────────────────────────────►

►──┬─┬─DEFine─┬─ ──OSA-osaname──┤ Define and Modify Options ├──────────┬───►◄
   │ └─MODify─┘                                                        │
   ├─┬─DELETE──┬─ ──OSA-osaname─────────────────────────────────────────┤
   │ ├─DISplay─┤                                                       │
   │ └─SWING───┘                                                       │
   ├─ADD── ──OSA-osaname── ──VIPA-vipaddr──┬────────────┬─┬───────────┬─┤
   │                                       └─ ──DEFAULT─┘ └─ ──STATIC─┘ │
   ├─REMove── ──VIPA-vipaddr────────────────────────────────────────────┤
   └─DATAFLOW───────────────────────────────────────────────────────────┘
```

Define and Modify Options:

```
├──┬──────────────────┬─┬──────────────────┬─┬───────────────────┬──────────►
   └─ ──READ-readsda──┘ └─ ──DATA-datasda──┘ └─ ──PORT-osaport───┘

   ┌─ ──LINK-0──────┐ ┌─ ──NET-GENET─┐ ┌─ ──MTU-1492─────┐
►──┼────────────────┼─┼──────────────┼─┼─────────────────┼─────────────────────►
   └─ ──LINK-link───┘ └─ ──NET-net───┘ └─ ──MTU-mtusize──┘

►──┬──────────────────────┬─┬──────────────────────────┬─────────────────────►
   └─ ──MASK-subnetmsk────┘ └─ ──IP-ipaddr──┬───────────┬┘
                                            └─ ──DEFAULT─┘

►──┬─────────────────────────────┬─┬─────────────────────────────┬──────────►
   └─ ──GATEWAY1-──┬─gate1─┬──────┘ └─ ──GATEWAY2-──┬─gate2─┬──────┘
                   └─NONE──┘                        └─NONE──┘

►──┬──────────────────────────────┬──────────────────────────────────────────┤
   └─ ──PRIMary-primosaname───────┘
```

DEFine
: defines an OSA-Express connection.

MODify
: changes the definitions for an OSA-Express connection. The name of the OSA-Express connection must already exist.

OSA-*osaname*
: specifies the name of the OSA-Express connection, where *osaname* is a 3- to 8-character alphanumeric connection name.

DELETE
: deletes an inactive OSA-Express connection.

DISplay
displays the defined parameters for a specific OSA-Express connection.

SWING
specifies that all the VIPAs associated with a specific OSA-Express connection can be moved to its alternate OSA-Express connection. You must specify the name of the active OSA-Express connection where VIPAs will be moved.

ADD
defines a VIPA for a primary OSA-Express connection and its associated alternate OSA-Express connection if an alternate is defined. The name that you specify for the OSA-Express connection must be the primary connection.

The real IP address and subnet mask of the OSA-Express connection must be defined before any VIPAs can be defined to that OSA-Express connection. The VIPA cannot reside in the same subnet as the networks to which the OSA-Express cards (primary and alternate) are connected.

VIPA-*vipaddr*
specifies a VIPA, where *vipaddr* is the numeric virtual IP address.

DEFAULT
specifies that the Internet Protocol (IP) address or VIPA being defined will be the default local IP address of the TPF system.

STATIC
specifies that the VIPA cannot be moved between processors. If you do not specify this parameter, the VIPA is movable.

REMOVE
removes the association of a specified VIPA from an OSA-Express connection and removes the VIPA definition from the TPF system. Both the primary OSA-Express connection and its associated alternate OSA-Express connection, if defined, must be inactive to remove a VIPA.

DATAFLOW
displays the number of packets sent and received across each OSA-Express connection in a 5-second interval .

READ-*readsda*
specifies the read control device of the OSA-Express connection, where *readsda* is the symbolic device address (SDA) of the read control device. The READ parameter also implicitly specifies the write control device, which is defined as the read control SDA plus 1.

DATA-*datasda*
specifies the data control device of the OSA-Express connection, where *datasda* is the SDA of the data control device.

PORT-*osaport*
specifies the name of the port on the OSA-Express card, where *osaport* is a 3- to 8-character alphanumeric port name. The name of the port must be the same for all hosts that have connections to the same OSA-Express card.

LINK-*link*
specifies the logical link number of the OSA-Express connection, where *link* is the logical link number. The default logical link is zero and is the only value that is supported.

NET-*net*
specifies the type of network to which the OSA-Express card is connected, where *net* is one of the following:

GENET
specifies a Gigabit Ethernet network.

FENET
specifies a Fast Ethernet network.

MTU-*mtusize*
specifies the maximum transmission unit (MTU) of the OSA-Express connection, where *mtusize* is a 576- to 3840-decimal value. The MTU parameter is ignored when defining or modifying an alternate OSA-Express connection. The MTU size for the alternate OSA-Express connection is automatically set to the same value as the MTU size of its associated primary OSA-Express connection. If the primary OSA-Express connection MTU size is changed, the alternate OSA-Express connection MTU size is also automatically changed to the same value.

MASK-*subnetmask*
defines the subnet mask of the network to which the OSA-Express card is connected, where *subnetmask* is the numeric subnet mask address. The subnet mask cannot be modified once it has been defined.

IP-*ipaddr*
defines the real IP address of the TPF host associated with the OSA-Express connection, where *ipaddr* is the numeric local IP address. The IP address cannot be modified once it has been defined.

GATEWAY1-*gate1*
specifies one of the two possible default gateways to be used as the first hop of the route for outbound packets that are sent by the TPF system across this OSA-Express connection, where:

gate1
is the numeric IP address of the gateway.

NONE
deletes previously defined information for the gateway.

GATEWAY2-*gate2*
specifies one of the two possible default gateways to be used as the first hop of the route for outbound packets that are sent by the TPF system across this OSA-Express connection, where:

gate2
is the numeric IP address of the gateway.

NONE
deletes previously defined information for the gateway.

PRIMary-*primosaname*
specifies the name of the primary OSA-Express connection for which this OSA-Express connection is the alternate, where *primosaname* is the primary OSA-Express connection name. Each primary connection can have only one alternate connection. You cannot modify the primary OSA-Express connection name once it has been defined.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZOSAE HELP
  ZOSAE ?
- New or changed definitions can be used immediately; an IPL is not required.

- To accommodate long ZOSAE command entries, you can first enter the ZOSAE command with the DEFINE parameter specified with some of the parameters; you can specify additional parameters on subsequent entries of the ZOSAE command with the MODIFY parameter specified.
- The only definitions you can change while the OSA-Express connection is active are MTU size and gateways.
- You must delete an alternate OSA-Express connection before you can delete its associated primary OSA-Express connection.
- If you define both the GATEWAY1 and GATEWAY2 parameters, these parameters alternate as the default gateway if both are active. You must define the IP address and subnet mask of the OSA-Express connection before you can define the default gateway. The IP address of the gateway must be in the same subnet as the network to which the OSA-Express card is connected.
- You must define the following parameters before you can activate an OSA-Express connection:
  - OSA
  - READ
  - DATA
  - PORT
  - MASK
  - IP.

Examples

In the following example, an OSA-Express connection is defined.

```
User:   ZOSAE DEF OSA-OSA1PRIM READ-E00 DATA-E02 PORT-OSAPORT1

System: OSAE0001I 08:14:31 OSA-OSA1PRIM DEFINED
```

In the following example, an IP address and subnet mask is defined to primary OSA-Express connection OSA1PRIM.

```
User:   ZOSAE MOD OSA-OSA1PRIM IP-9.117.241.30 MASK-255.255.255.0

System: OSAE0002I 08:14:31 OSA-OSA1PRIM MODIFIED
```

In the following example, an alternate OSA-Express definition OSA1BACK is defined to the primary OSA-Express connection OSA1PRIM.

```
User:   ZOSAE DEF OSA-OSA1BACK PRIMARY-OSA1PRIM

System: OSAE0001I 08:14:31 OSA-OSA1BACK DEFINED
```

In the following example, an OSA-Express definition is deleted.

```
User:   ZOSAE DELETE OSA-OSA1PRIM

System: OSAE0003I 08:14:31 OSA-OSA1PRIM DELETED
```

In the following example, a static VIPA is added to an OSA-Express definition.

```
User:   ZOSAE ADD OSA-OSA1PRIM VIPA-9.117.251.40 STATIC

System: OSAE0004I 08:14:31 VIPA-9.117.251.40 DEFINED TO OSA-OSA1PRIM
```

In the following example, a VIPA is removed from the TPF system.

```
User:   ZOSAE REMOVE VIPA-9.117.251.40

System: OSAE0005I 08:14:31 VIPA-9.117.251.40 DELETED
```

In the following example, VIPAs swing from a primary OSA-Express connection OSA1PRIM to its alternate OSA-Express connection OSA1BACK.

```
User:   ZOSAE SWING OSA-OSA1BACK

System: OSAE0006I 08:14:31 VIPAS ON OSA-OSA1PRIM SWUNG TO OSA-OSA1BACK
```

In the following example, information is displayed for a primary OSA-Express connection.

```
User:   ZOSAE DISPLAY OSA-OSAPRIM1

System: OSAE0007I 15.18.32 BEGIN OSA DISPLAY

        NAME - OSAPRIM1                REAL IP - 111.111.111.050      READ - 0200
        TYPE - PRIMARY                    MASK - 255.255.255.000     WRITE - 0201
         ALT - OSABACK1               GATEWAY1 - 111.111.111.032      DATA - 0205
        PORT - OSAPORT1               GATEWAY2 - 111.111.111.033
        LINK - 0
         NET - GENET                  CURRENT STATUS - ACTIVE
         MTU - 1492                   DESIRED STATUS - ACTIVE

        VIPAS CURRENTLY ASSIGNED TO OSAPRIM1:
            111.117.249.050      MOVABLE
            111.117.249.053      STATIC
            111.117.249.088      MOVABLE, NOT OWNED BY THIS CPU

        END OF DISPLAY
```

In the following example, information is displayed for the alternate OSA-Express connection.

```
User:   ZOSAE DISPLAY OSA-OSABACK1

System: OSAE0007I 15.18.32 BEGIN OSA DISPLAY

        NAME - OSABACK1                REAL IP - 111.111.117.033      READ - 0420
        TYPE - ALTERNATE                  MASK - 255.255.255.000     WRITE - 0421
        PRIM - OSAPRIM1               GATEWAY1 - 111.111.117.050      DATA - 0426
        PORT - OSAPORT2               GATEWAY2 - NONE
        LINK - 0
         NET - GENET                  CURRENT STATUS - ACTIVE
         MTU - 1492                   DESIRED STATUS - ACTIVE

        VIPAS CURRENTLY ASSIGNED TO OSAPRIM1:
            111.117.249.050      MOVABLE
            111.117.249.053      STATIC
            111.117.249.088      MOVABLE, NOT OWNED BY THIS CPU

        END OF DISPLAY
```

In the following example, OSA data flow statistics are displayed.

```
User:   ZOSAE DATAFLOW

System: OSAE0008I 17.25.49 BEGIN PROCESSING OSA DATAFLOW STATISTICS
        CSMP0097I 17.25.54 CPU-B SS-BSS  SSU-HPN  IS-01
        OSAE0009I 17.25.54 OSA DATAFLOW STATISTICS FOR A 5-SECOND INTERVAL

                OSA        PACKETS     PACKETS
                           SENT        RECEIVED
                --------   ---------   ----------
                OSA1B            262          263
                OSA2B            284          285
                OSA3B              0            1
        END OF ZOSAE DATAFLOW DISPLAY
```

Related Information

See the *TPF Transmission Control Protocol/Internet Protocol* for more information about IP routers, TCP/IP native stack support, and OSA-Express support.

ZPAGE–Continue Output

Use this command to continue displaying the output of a command when there is more information than can be displayed on the screen at one time. When this situation occurs, the following message is displayed:

```
MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE
```

Requirements and Restrictions

- You must enter the ZPAGE command within 1 minute after you are prompted or the remaining data is no longer available.
- If you enter another command before you enter the ZPAGE command and that command also prompts you to enter the ZPAGE command, the remaining data from the first command is no longer available. When you enter the ZPAGE command, only the remaining data from the second command is displayed.

Format

```
►►──ZPAGE──────────────────────────────────────────────────────────►◄
```

Additional Information

- You can enter other commands before you enter the ZPAGE command.
- You can change the amount of data that is displayed on the screen before you are prompted to enter the ZPAGE command. To change the amount of data, use the UOP3 user exit.

Examples

In the following example, there is more information than can be displayed on the screen at one time in the following example. You must enter the ZPAGE command to display the remaining data.

```
User:   ZOLDR DISP ALL

System: OLDR1220I CLE3 - BEGIN DISPLAY
                  LOADSET      STATUS
                  FIX1017      CREATED ON 01/13/94 10.34.08
                               INACTIVE

                  FIX1018      CREATED ON 01/13/94 20.42.17
                               ACTIVATED ON CPUID B ON 01/13/94 21.03.51
                               ACTIVATED ON CPUID C ON 01/13/94 02.52.12
                  MORE DATA AVAILABLE, ENTER ZPAGE TO CONTINUE

User:   ZPAGE

System: OLDR1224I CLDN - LOADSET INFORMATION CONTINUED
                  LOADSET      STATUS
                  LLTPKG       CREATED ON 01/13/94 09.10.26
                               SELECTIVELY ACTIVATED ON CPUID B ON 01/13/94 10.34.08

                  XX2RF        CREATED ON 01/13/94 10.31.01
                               PENDING DELETION

                  YLD07        CREATED ON 01/13/94 17.13.52
                               ACCEPT IN PROGRESS

        CLE3 - END OF DISPLAY
```

Related Information

See *TPF System Installation Support Reference* for more information about the UOP3 user exit.

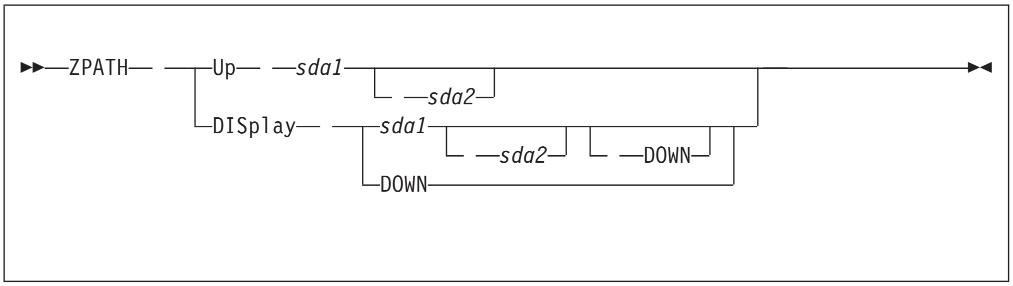
ZPATH–Path Management

Use this command to:
- Display the status of paths associated with a DASD
- Bring online all available paths associated with a DASD.

Requirements and Restrictions

If you specify the UP parameter, you must wait for the ZPATH function to complete successfully before you can enter the ZPATH command again with the UP parameter.

Format



Up
: attempts to bring all the paths online that are associated with the specified DASD.

DISplay
: displays the status of all the paths that are associated with the specified DASD.

DOWN
: displays the status of all the paths that are not online and associated with the specified DASD.

sda1
: is the symbolic device address of a DASD.

sda2
: is a symbolic device address greater than or equal to *sda1*. Use this variable to specify a range of DASD.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZPATH HELP
  ZPATH ?
- This command displays the current status of the paths. This status is not verified.

Examples

All the available paths in the specified DASD range are brought online in the following example.

```
User:   ZPATH UP 200 500

System: PATH0010I 09.29.51 ZPATH PROCESS COMPLETED ON VALID DASD SDA
```

The status of the paths for device 4E0 is displayed in the following example.

3E, 3A, and 2C are the paths to device 4E0. The status of each path is shown by U (up) or D (down).

```
User:   ZPATH DIS 4E0

System: PATH0010I 09.31.44 ZPATH PROCESS COMPLETED ON VALID DASD SDA
        CSMP0097I 09.31.44 CPU-B SS-BSS  SSU-HPN  IS-01
        PATH0015I 09.31.44 STATUS DISPLAY FOLLOWS
        DEVICE                STATUS OF PATHS
        ------     ------------------------------------------------
        04E0        3E(U)  3A(U)  2C(D)
        PATH0029I 09.31.44 END OF DISPLAY
```

Related Information

None.

ZPLMT–Purge LMT/OMT Queue

Use this command to remove output messages that were queued on file by the long message transmitter (LMT) program or the output message transmitter (OMT) program.

Requirements and Restrictions

None.

Format

```
►►──ZPLMT── ──┬─lniata─┬──┬───┬──────────────────────────────────────►◄
              └─leid───┘  └─N─┘
```

lniata
  is a line number, interchange address, and terminal address.

leid
  is a logical end-point identifier.

N purges messages without saving them to a PLM tape.

Additional Information

You can also enter the ZNPRG command to remove output messages that were queued on file for a particular node.

Examples

In the following example, output messages that were queued on file are removed and saved to a general tape.

```
User:   ZPLMT 000000

System: PLMT0002I 14.25.06 QUEUE FOR 000000 PURGED TO TAPE PLM, MSG CNT -      2
        COTC0080A 14.25.06 TCLS HPN    REMOVE PLM FROM DEVICE 421
```

Related Information

See *TPF Data Communications Services Reference* or *TPF ACF/SNA Data Communications Reference* for more information about the long message transmitter program.

ZPMIG–Migrate Pool Structures

Use this command to:

- Convert the pool data structures for each subsystem from pool expansion (PXP) format to 32-way loosely coupled pool support format
- Return the pool data structures for each subsystem from 32-way loosely coupled pool support format back to pool expansion (PXP) format
- Display the migration state of each processor in the complex.

Requirements and Restrictions

- You can enter this command only from a processor where the pool format provided by 32-way loosely coupled pool support is installed and active.
- You can enter this command only in 1052 state or higher.

Format

```
►►──ZPMIG── ──┬─CONVERT──┬──────────────────────────────────────────────►◄
              ├─FALLBACK─┤
              └─STATUS───┘
```

CONVERT
converts the pool data structures for the current subsystem from PXP format to 32-way loosely coupled pool support format. When pool conversion is completed, all pool data structures on DASD are in 32-way loosely coupled pool support format.

Notes:

1. If your complex is loosely coupled, you must ensure that the pool data structures in main processor storage on each processor in the complex are current before specifying the CONVERT parameter. Do this by entering the ZDFPC command on each active processor in the complex.
2. All processors in the complex must be currently running from an image containing 32-way loosely coupled pool support before you convert the pool data structures. To ensure that all active processors are using 32-way loosely coupled pool support, enter the ZPMIG command with the STATUS parameter specified from each active processor. If message PMIG0011I is returned, the processor is using 32-way loosely coupled pool support. If any other message is returned, the processor is not using 32-way loosely coupled pool support. You must re-IPL the processor using an image containing 32-way loosely coupled pool support before you can specify the CONVERT parameter.
3. You need to enter the command with the CONVERT parameter specified once for each subsystem in the loosely coupled complex. The command can be entered from any active processor in the complex. After processing is successfully completed, all pool data structures that reside in processor storage and on DASD for the subsystem are in 32-way loosely coupled pool support format.

FALLBACK
changes the pool data structures on DASD for each subsystem from 32-way loosely coupled pool support format back to PXP format.

STATUS
displays the migration state for each processor in the complex. The migration state can be MIGRATED, UNMIGRATED, or INACTIVE.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZPMIG HELP
ZPMIG ?

Examples

In the following example, pool data structures are converted from PXP format to 32-way loosely coupled pool support format.

```
User:   ZPMIG CONVERT

System: PMIG0001I 13.31.29 POOL CONVERSION COMPLETE
```

In the following example, pool data structures are changed from 32-way loosely coupled pool support format back to PXP format.

```
User:   ZPMIG FALLBACK

System: PMIG0002I 13.31.29 POOL CONVERSION FALLBACK COMPLETED
```

The migration state for each processor defined in the complex is displayed in the following example.

```
User:   ZPMIG STATUS

System: PMIG0011I 13.26.36 32-WAY LC POOL MIGRATION STATUS
        THE POOL DATA STRUCTURES ARE NOT CONVERTED
            PROCESSOR        STATUS
            ---------      ----------
                B          MIGRATED
                C          UNMIGRATED
                D          UNMIGRATED
                E          INACTIVE
        END OF DISPLAY+
```

Related Information

See the *TPF Migration Guide: Program Update Tapes* for more information about converting the entire complex from the old pool format to the new pool format.

ZPOOL DISPLAY–Display the OPMAAA or OPMBBB Sections

Use this command to display the online pool maintenance available (OPMAAA) sections or the online pool maintenance deactivation (OPMBBB) sections, sorted by pool type.

Requirements and Restrictions

None.

Format

```
►►──ZPOOL DISPLAY── ──┬─OPM──────────────────┬──────────────────────────►◄
                      │         ┌─ ──CUR─┐   │
                      ├─OPMAAA──┼────────┼───┤
                      │         └─ ──NEW─┘   │
                      │         ┌─ ──CUR─┐   │
                      └─OPMBBB──┼────────┼───┘
                                └─ ──NEW─┘
```

OPM
displays the current copy of the OPMAAA sections.

OPMAAA
displays the OPMAAA sections, which contains the areas of file storage that are used by the online system as random pool storage.

OPMBBB
displays the OPMBBB sections, which contain the areas of pool storage that are to be deactivated.

CUR
displays the current copy of the OPMAAA or OPMBBB sections.

Note: The current copy of the OPMAAA or OPMBBB sections becomes the new copy after ZPOOL GENERATION UPDATE processing ends successfully.

NEW
displays the new copy of the OPMAAA or OPMBBB sections.

Note: The new copy of the OPMAAA or OPMBBB sections becomes the current copy after ZPOOL GENERATION UPDATE processing ends successfully.

Additional Information

None.

Examples

The following example displays the current OPMAAA sections.

```
User:   ZPOOL DISPLAY OPM

System: B1AA0004I 14.16.03 RECOUP - CUR OPMAAA

        STARTING LOW  HIGH    POOL               BASE             NUMBER OF
        CCCC/HH  MOD. MOD.    TYPE               ORDINAL          RECORDS
        -------  ---- ----  --------          ----------        ----------
         657/ 0     0  127  SST-DEVA                   0            101760
         700/ 0     0   63  SDP-DEVA            50331648           1475520
         652/ 0     0  127  LST-DEVA                   0            115200
           1/ 0     0   63  LDP-DEVA            50331648           2880000
         101/ 0     0   63  LDP-DEVA            53211648           1612800
         569/ 0     2  127  4LT-DEVA            59768832              6300
         569/ 5     2  127  4LT-DEVA            59775132              6300
         572/ 0     2  127  4ST-DEVA            58720256              6300
         572/ 5     2  127  4ST-DEVA            58726556              6300
         348/ 0     0   63  4DP-DEVA            58806656             67200
         355/ 0     0   63  4DP-DEVA            67102864             19200
         498/ 0     0   63  4DP-DEVA           134198526             19200
         560/ 0     0   63  4DP-DEVA            58720256             86400
         854/ 0     0   63  4DP-DEVA            58873856            288000
         289/ 0     0   63  4DP-DEVA            59161856            192000
         885/ 0     0   63  4DP-DEVA            59353856            960000
        1145/ 0     0   63  4D6-DEVA            59353856          28560000
        2340/ 0     0   63  4D6-DEVA         14829039446         394803000

        B1AA0011I 14.16.03 END OF DISPLAY
```

The following example displays the current OPMBBB sections.

```
User:   ZPOOL DISPLAY OPMBBB

System: B1AA0004I 14.16.03 RECOUP - CUR OPMBBB
        STARTING LOW  HIGH    POOL               BASE             NUMBER OF
        CCCC/HH  MOD. MOD.    TYPE              ORDINAL            RECORDS
        -------  ---- ----  --------          ----------        ----------
         101/ 0     0   63  LDP-DEVA            53211648           1612800
        B1AA0011I 14.16.03 END OF DISPLAY
```

The following example displays the new OPMAAA sections.

```
User:   ZPOOL DISPLAY OPMAAA NEW

System: B1AA0004I 14.16.03 RECOUP - NEW OPMAAA

        STARTING LOW  HIGH    POOL              BASE            NUMBER OF
        CCCC/HH  MOD. MOD.    TYPE              ORDINAL         RECORDS
        -------  ---- ----  --------          ----------        ----------
         657/ 0     0  127  SST-DEVA                   0           101760
         700/ 0     0   63  SDP-DEVA            50331648          1475520
         652/ 0     0  127  LST-DEVA                   0           115200
           1/ 0     0   63  LDP-DEVA            50331648          2880000
         101/ 0     0   63  LDP-DEVA            53211648          1612800
         569/ 0     2  127  4LT-DEVA            59768832             6300
         569/ 5     2  127  4LT-DEVA            59775132             6300
         572/ 0     2  127  4ST-DEVA            58720256             6300
         572/ 5     2  127  4ST-DEVA            58726556             6300
         348/ 0     0   63  4DP-DEVA            58806656            67200
         355/ 0     0   63  4DP-DEVA            67102864            19200
         498/ 0     0   63  4DP-DEVA           134198526            19200
         560/ 0     0   63  4DP-DEVA            58720256            86400
         854/ 0     0   63  4DP-DEVA            58873856           288000
         289/ 0     0   63  4DP-DEVA            59161856           192000
         885/ 0     0   63  4DP-DEVA            59353856           960000
        1145/ 0     0   63  4D6-DEVA            59353856         28560000
        2340/ 0     0   63  4D6-DEVA         14829039446        394803000

        B1AA0011I 14.16.03 END OF DISPLAY
```

The following example displays the new OPMBBB sections.

```
User:   ZPOOL DISPLAY OPMBBB NEW

System: B1AA0004I 14.16.03 RECOUP - NEW OPMBBB
        STARTING LOW  HIGH    POOL              BASE            NUMBER OF
        CCCC/HH  MOD. MOD.    TYPE             ORDINAL           RECORDS
        -------  ---- ----  --------         ----------         ----------
         101/ 0     0   63  LDP-DEVA            53211648          1612800
        B1AA0011I 14.16.03 END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZPOOL GENERATION–Pool Generation and Reallocation Functions

Use this command to create new directory records or change existing directory records to conform to a changed file layout.

Requirements and Restrictions

- You must convert all pool structures to 32-way loosely coupled pool support format before using this command. See "ZPMIG–Migrate Pool Structures" on page 1116 for more information about converting pool structures.
- After you enter the RECONFIGURE, UPDATE, or FALLBACK parameter, the TPF system will prompt you to continue or abort. If you continue, the TPF system will receive a catastrophic I00003C system error and an IPL will occur.
- The CREATE parameter can be specified when all active processors are in 1052 or NORM state (all active processors must be in the same state). All other parameters can be specified only when all active processors are in 1052 state.

Format

```
►►──ZPOOL GENERATION── ──┬─CREATE───────────────────┬──────────────────────────►◄
                         ├─RECONFIGURE──────────────┤
                         ├─UPDATE───────────────────┤
                         ├─FALLBACK─────────────────┤
                         ├─ONLINE── ──┬─ABORT────┬──┤
                         │            └─CONTINUE─┘  │
                         └─INIT─────────────────────┘
```

CREATE
creates initial pool directories or changes existing pool directories from FACE table (FCTB) generation input.

Notes:

1. To preserve and merge existing pool directories into the newly created pool directories, enter **ZPOOL GENERATION RECONFIGURE** after you enter ZPOOL GENERATION CREATE. If you do not, all existing pool files are made available to the TPF system.
2. Enter **ZPOOL GENERATION UPDATE** to make the newly created pool directories active.

RECONFIGURE
preserves and merges existing pool directories into the newly created pool directories.

Note: You can specify this parameter only after you enter ZPOOL GENERATION CREATE.

UPDATE
rolls new pool directories into the pool rollin directory (#SONRI) and updates keypoint 9 (CTK9).

Note: The current pool rollin directory (#SONRI) is first saved to the recoup SONRI save area (#SONSV) so it can later be used to restore the #SONRI if a problem requires you to do so.

FALLBACK
rolls the recoup SONRI save area (#SONSV) into the pool rollin directory (#SONRI) and updates CTK9.

ONLINE
specifies an action, where:

ABORT
forces ZPOOL GENERATION UPDATE or ZPOOL GENERATION FALLBACK processing to end without completing.

CONTINUE
continues ZPOOL GENERATION UPDATE or ZPOOL GENERATION FALLBACK processing.

INIT
creates the current pool segment table (#PSTXCUR).

Notes:

1. Only specify this parameter when migrating to integrated online pool maintenance and recoup support. See *TPF Migration Guide: Program Update Tapes* for more information.
2. Ensure that all pool segments in the FACE table (FCTB) are allocated. If you specify this parameter and there are pool segments in the FCTB that are not allocated, the #PSTXCUR and #PSTXNEW records that are built will not match the current pool configuration.

Additional Information

None.

Examples

The following example creates the information necessary for pool generation and reallocation.

```
User:   ZPOOL GENERATION CREATE

System: DYD30001I 05.39.34 NEW POOL DIRS CREATED
```

The following example preserves and merges existing pool directories into newly created pool directories.

```
User:   ZPOOL GENERATION RECONFIGURE

System: DYD40001I 05.39.34 ZPOOL GENERATION RECONFIGURE COMPLETED
```

The following example rolls new pool directories into the pool rollin directory (#SONRI) and updates keypoint 9 (CTK9).

```
User:   ZPOOL GENERATION UPDATE

System: DYD60002I 09.09.29 --- WAITING ON ENTRY: ZPOOL GENERATION ONLINE ABORT OR CONTINUE

User:   ZPOOL GENERATION ONLINE CONTINUE

        CPSE0053T 09.09.29 IS-0001 SS-BSS  SSU-HPN  SE-002677 OPR-I00003C CATASTROPHIC
        010000B CYE040  000000EA
        SYSTEM STATE DISABLED
        STATE CHANGE DISABLED
        PREVIOUS ERRORS ENCOUNTERED
        2677 OPR-I00003C
        CPSE0011I 09.09.29 OPR-I00003C DUMP IN PROGRESS, LOCATION 09E0FF40
        CPSF0005I 09.09.29 RECOVERY TO 1052 STATE ONLY
        CPSF0005I 09.09.29 RECOVERY TO 1052 STATE ONLY
        DYDG0003I 09.09.29 -POOL GEN COMPLETE
        CPSF0010I 09.09.29 DISK QUEUES PROCESSED
        CPSF0014W 09.09.29 CRITICAL RECORD FILING COMPLETED
        CPSF0013I 09.09.29 SOFTWARE IPL INITIATED
```

The following example creates the current pool segment table (#PSTXCUR) when migrating to integrated online pool maintenance and recoup support.

```
User:    ZPOOL GENERATION INIT

System:  CSMP0097I 13.58.49 CPU-B SS-BSS  SSU-HPN  IS-01
         DYDA0001I 13.58.49 PSTXCUR HAS BEEN CREATED
         CSMP0097I 13.58.49 CPU-B SS-BSS  SSU-HPN  IS-02
         BBLD0001I 13.58.49 IPART INITIALIZATION COMPLETED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZPOOL INIT–Migration and Initialization Package

Use this command to initialize pool directory update (PDU), pseudo directory (PSDIR), and recoup records.

Requirements and Restrictions

- You must install APAR PJ27354 on each processor in your TPF system before you enter this command.
- All associated records must be allocated for each subsystem.

Format

```
►►──ZPOOL INIT── ──┬─PDU────┬──────────────────────────────────────────►◄
                   ├─PSDIR──┤
                   └─RECOUP─┘
```

PDU
: initializes all records required for running an online PDU, and completes pool directory migration for all directory records. This also initializes pool section tables and pool release chain structures. Specify this parameter only one time to initialize new records.

PSDIR
: initializes pseudo directory records, which are required for recoup or PDU processing, and completes pool directory migration for all directory records.

RECOUP
: initializes all records required for running online recoup. This includes descriptors and control recoup, and completes pool directory migration for all directory records. Specify this parameter only one time to initialize new records.

Additional Information

The pool directory migration performed by the PDU, PSDIR, and RECOUP parameters, migrates pool directories from TPF 3.1 format to TPF 4.1 format and sets the CY3BON and CY3DON fields to zero. See *TPF Migration Guide: TPF 3.1 System to TPF 4.1 System* for more information about migration considerations for pools.

The following table lists the records that are initialized for each ZPOOL INIT command parameter.

Record Type	FACE Type	RECOUP	PDU	PSDIR
#$C2EC	#IBMML		X	
#BKDCTL	#IBMM4	X		
#BKMST	#IBMM4	X		
#BKWRK	#IBMM4	X		
#BREATB8	#BREATB8	X		
#BRHIST8	#BRHIST8	X		
#BRIDCOR	#BRIDCOR	X		

Record Type	FACE Type	RECOUP	PDU	PSDIR
#BRIDDE8	#BRIDDE8	X		
#BRIDSA8	#BRIDSA8	X		
#BRIDTB8	#BRIDTB8	X		
#BRIDTO8	#BRIDTO8	X		
#BRLOTB8	#BRLOTB8	X		
#C414C	#IBMM4	X	X	
#C414CX	#IBMM4	X	X	
#EXCTBL	#IBMM4		X	
#E81E8	#IBMM4	X		
#KY9CPY1	#IBMM4	X		
#L81L8	#IBMM4	X		
#MPRECP	#MPRECP	X		
#ORHRC	#IBMM4	X		
#PSTXCUR	#PSTXCUR		X	
#PSTXNEW	#PSTXNEW		X	
#RCPST1	#IBMMS	X		
#RC8RFS	#RC8RFS		X	
#SONCP	#SONCP	X		X
#SONDE	#SONDE	X		X
#SONROLL	#SONROLL	X		X
#SONRPE	#SONRPE	X		X
#SONRPM	#SONRPM	X		X
#SONSKP	#SONSKP	X	X	X
#SONSV	#SONSV	X		X
#SONUP	#SONUP	X	X	X
#SRHH1P	#SRHH1P	X		
#SRM31A8	#SRM31A8	X		
#SRM41A8	#SRM41A8	X		
#SRM51A8	#SRM51A8	X		

Examples

The following example initializes all records required for running an online pool directory update (PDU).

```
User:   ZPOOL INIT PDU

System: BRV20000I 09.09.29 PDU INIT START
        BRV20002I 09.09.29 INIT COMPLETED
```

The following example initializes pseudo directory records, which are required for recoup or pool directory update (PDU).

```
User:   ZPOOL INIT PSDIR

System: BRV20003I 09.09.29 PSEUDO DIR INIT START
        BRV20002I 09.09.29 INIT COMPLETED
```

The following example initializes all records required for running online recoup.

```
User:   ZPOOL INIT RECOUP

System: BRV20001I 09.09.29 RECOUP INIT START
        BRV20002I 09.09.29 INIT COMPLETED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZPOOL RPFS–Reset Pool Function Switches

Use this command to reset the pool function switch when you cannot start a pool function because of an irrecoverable condition. Each time you start a pool utility (for example, recoup), the pool function switch is set to prevent you from starting another pool utility (for example, pool directory update (PDU)). Each pool utility should reset the pool function switch when it is completed successfully or unsuccessfully. If the pool utility fails to reset the pool function switch because of an irrecoverable error, reset the pool function switch to allow another pool utility to run.

Requirements and Restrictions

None.

Format

```
►►──ZPOOL RPFS──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The file pool function switches are reset in the following example.

```
User:   ZPOOL RPFS

System: CYAE0005I 09.40.59 POOL RPFS-PERFORMED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZPROT–Update the Utility PROT

Use this command to:

- Add an entry to the utility processor resource ownership table (PROT)
- Delete an entry from the utility PROT
- Assign ownership of a utility to a subsystem on a processor
- Release ownership of a utility from a subsystem on a processor.

Requirements and Restrictions

- You must enter this command from each active processor where you want to add or delete the utility.
- Before you can assign ownership of a utility, you must add an entry to the utility PROT for that utility.

Format

```
►►──ZPROT── ──┬─ADD─┬── ──UT── ──utility── ──ssname──────────────────────────►◄
              ├─DEL─┤
              ├─ASN─┤
              └─REL─┘
```

ADD
adds an entry to the PROT.

DEL
deletes an entry from the PROT.

ASN
assigns ownership of a utility to a subsystem.

REL
releases ownership of a utility from a subsystem.

utility
is the 4-character alphanumeric name of a system utility.

ssname
is the 1- to 4-character alphanumeric name of a subsystem.

Additional Information

- Before deleting an entry, enter this command and specify the REL parameter to mark the entry as not in use. Then, enter this command again and specify the DEL parameter to remove the entry from the utility PROT.
- You can use this command to delete a utility on another processor if that processor is not active.
- You can add entries to the utility PROT for utilities with the same name provided as long as the utilities have different subsystem names.
- A subsystem name is not verified until you try to assign ownership using the ZPROT ASN command.

Examples

The following example adds the POOL utility to the utility PROT for the basic subsystem (BSS) on the current processor.

```
User:   ZPROT ADD UT POOL BSS

System: PROT0051A 14.26.27  PROT ASN UT POOL BSS B ASSIGNED
```

Related Information

- See *TPF Main Supervisor Reference* for more information about the processor resource ownership table (PROT).
- See “ZPROT DSP–Display Ownership” on page 1130 for information about how to display processor resource ownership information.

ZPROT DSP–Display Ownership

Use this command to display the following information:

- Names of the tape devices or system utilities owned by the current processor
- Names of the tape devices or system utilities owned by each processor
- Names of the processors that own specific tape devices or system utilities.

Requirements and Restrictions

None.

Format

```
►►──ZPROT DSP── ──┬─TP──┬──────────────────┬─┬──────────────────────►◄
                  │     ├─ ──ALL───────────┤ │
                  │     └─ ──addr1──┬──────┬┘ │
                  │                 └-addr2┘  │
                  └─UT──┬───────────────────┬─┘
                        ├─ ──ALL────────────┤
                        └─ ──utility── ──ssname─┘
```

TP
: displays ownership information about tape devices.

UT
: displays ownership information about system utilities.

ALL
: displays the names of the tape devices or system utilities owned by each processor.

addr1
: displays the names of the processors that own a specific tape device, where *addr1* is the hexadecimal address of the tape device from X'00'–X'FFF'.

addr2
: displays the names of the processors that own the tape devices in the specified address range, where *addr2* is the end of the address range and *addr1* is the beginning of the address range.

utility
: displays the names of the processors that own a specific system utility, where *utility* is the 4-character alphanumeric name of the system utility.

ssname
: is the 1- to 4-character alphanumeric name of a subsystem.

Additional Information

- Enter **ZPROT DSP TP** or **ZPROT DSP UT** to display the tape devices or system utilities owned by the current processor.

Examples

The following example displays the names of the tape devices owned by the current processor.

```
User:   ZPROT DSP TP

System: PROT0052I 13.39.46 PROT DSP TP

        PROC B
        420 422 421
```

The following example displays the names of the processors that own the tape devices at address 420 through address 425. An asterisk (*) indicates that an entry was not created in the utility PROT for a tape device. NONE indicates that the tape control unit is online, but a tape device was not assigned ownership.

```
User:   ZPROT DSP TP 420-425

System: PROT0052I 13.39.46 PROT DSP TP 420-425

        420 B 421 B 422 B 423 * 424 * 425 NONE

        * - DEVICE ADRS NOT IN PROT
```

The following example displays the name of the processor that owns the POOL utility assigned to the basic subsystem (BSS).

```
User:   ZPROT DSP UT POOL BSS

System: PROT0050I 13.39.46 PROT DSP UT POOL BSS
        PROC B
```

Related Information

- See *TPF Main Supervisor Reference* for more information about the processor resource ownership table (PROT).
- See “ZPROT–Update the Utility PROT” on page 1128 for information about how to update the PROT.

ZPSMS–Processor Status Management Services

Use this command to do the following:

- Deactivate a processor
- Enable or disable cycling
- Activate or deactivate a subsystem
- Display status information about the processors in a loosely coupled complex or about the subsystems in a multiple database function (MDBF) environment.

Requirements and Restrictions

None.

Format

```
>>--ZPSMS-------------------------------------------------------------------->

>--+-PRocessor--+---------+--DEACtivate--id--+------------+-+--------------><
   |            '--FORCE--'                  '--REServed--' |
   +-Cycle--+-ENable--+------+-+-----------------------------+
   |        |         '--BP--' |                             |
   |        '-DISable----------'                             |
   +-SS--+-ACTivate---+--ssname------------------------------+
   |     '-DEACtivate-'                                      |
   '-Display--+-PRocessor-+----------------------------------'
              +-MDBF------+
              '-PRSS------'
```

PRocessor
deactivates a processor.

FORCE
removes a processor from the loosely coupled complex. All locks held by the processor are cleared, all appropriate tables and keypoints are updated to show the processor is inactive, and all other processors in the complex are notified of the new status. Interprocessor communication (IPC) to the deactivated processor stops. The processor from which the PROC FORCE DEACTIVATE parameters are specified will take over the recovery log of the processor that will be deactivated before its locks are released.

Attention: If the processor that is being deactivated by force is the current processor, the status of the processor is not checked before the processor is deactivated. For example, the processor may be deactivated if the TPF system is not in 1052 state, if active tapes are mounted, or if there are active ECBs in the TPF system.

If the processor that is being deactivated by force is not the current processor, it is assumed that the processor is inactive and cannot respond. If this is not the case, the integrity of the complex is jeopardized.

DEACtivate *id*
deactivates a processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

REServed
reserves the CPU ID for a processor. That is, the CPU ID is still associated with the CPU serial number. Otherwise, the CPU ID can be allocated to another CPU serial number.

Cycle
enables or disables cycling.

ENable
enables cycling if cycling was disabled using this command.

BP
overrides all indicators set by the TPF system that prevent cycling. If you do not specify the BP parameter, cycling can be enabled only if it was disabled using this command.

DISable
disables cycling.

SS
activates or deactivates a subsystem.

ACTivate
activates a subsystem.

DEACtivate
deactivates a subsystem.

ssname
is the 1- to 4-character name of a subsystem.

Display
displays status information about processors or subsystems.

PRocessor
displays status information about all the processors in the loosely coupled complex.

MDBF
displays status information about all the subsystems on this processor.

PRSS
displays status information about this subsystem in the loosely coupled complex.

Additional Information

Enter **ZPSMS CYCLE ENABLE BP** to enable cycling if cycling was disabled by the TPF system because of a system error.

Examples

The following example displays status information about all the processors in the loosely coupled complex.

```
User:   ZPSMS DISPLAY PR

System: PSMS0000I 14.26.27 CPUORD  CPUID       SERIAL     PIDT   CTKI

        PSMS0000I 14.26.27   00      B   FF02041030900000 1052   ACTIVE
        PSMS0000I 14.26.27   01      C   0000000000000000 REST   INACTIVE
        PSMS0000I 14.26.27   02      D   0000000000000000 REST   INACTIVE
        PSMS0000I 14.26.27   03      E   0000000000000000 REST   INACTIVE
        PSMS0000I 14.26.27   04      Z   0000000000000000 REST   INACTIVE
        PSMS0000I 14.26.27   05      0   0000000000000000 REST   INACTIVE
        PSMS0000I 14.26.27 END OF DISPLAY
```

Cycling is disabled in the following example.

```
User:   ZPSMS CYCLE DIS

System: PSMS0011I 12.26.01 CYCLING INHIBITED
```

Related Information

See *TPF Main Supervisor Reference* for more information about processor status management services (PSMS).

ZPTAP–Print Tape Status

Use this command to print on an available printer the status of all the configured tape devices.

Requirements and Restrictions

None.

Format

```
►►──ZPTAP────────────────────────────────────────────────────────────►◄
```

Additional Information

The format of the status information that is printed is the same as the output produced by the ZDTAP command.

Examples

In the following example, the status of all configured tape devices is printed on an available printer.

```
User:   ZPTAP

System: CFDT0001I 10.13.50 PTAP      - TAPE STATUS SENT TO PRINTER
```

Related Information

None.

ZPTCH–Maintain Memory Patch Decks

Use this command to simultaneously enter the following commands to change system virtual memory (SVM), also known as core memory:

- ZACOR
- ZADCA
- ZAPGM.

To change SVM, you must first place the TPF system in patch mode. Then you can build a patch deck of as many as 50 commands. As many as 20 patch decks can exist at one time. After you create a patch deck, you can take the TPF system out of patch mode and run the patch deck to make multiple changes to SVM simultaneously.

Requirements and Restrictions

The ZPTCH facility maintains patch decks in pool file records. Therefore, the TPF system must be cycled above 1052 state at least once so that ZPTCH restart can retrieve the pool file addresses.

Format

```
>>──ZPTCH── ──┬─┬─Begin─┬─ ──Name-patchname──────────────┬─────────────────><
              │ └─Purge─┘                                 │
              ├─Stop──────────────────────────────────────┤
              │                                  ┌─ ──LIST───┐ │
              ├─Execute── ──Name-patchname──┼───────────┼─┤
              │                                  └─ ──NOLIST─┘ │
              ├─Display── ──Name-─┬─patchname─┬──────────┤
              │                    └─ALL───────┘           │
              └─Remove── ──Name-patchname──┬──────────────┬┘
                                           └─ ──Entry-nn─┘
```

Begin
: places the TPF system in patch mode so you can create a new patch deck or activate an existing patch deck for message queuing. All subsequent ZACOR, ZADCA, and ZAPGM messages entered from this terminal are queued in the patch deck.

Purge
: deletes the specified patch deck.

Name-*patchname*
: is the 1- to 6-character alphanumeric patch deck name. Do not use ALL as a patch deck name.

Stop
: ends queuing on the current patch deck.

Execute
: runs the specified patch deck.

LIST
: lists each change included in the patch deck before running the patch deck.

NOLIST
runs the patch deck without listing the changes.

Display
displays the entries in the specified patch deck.

ALL
displays a list of all the patch decks.

Remove
removes an entry from the specified patch deck.

Entry-*nn*
is the 1- or 2-digit entry number in the patch deck to be removed. If you do not specify an entry number, the default is the last entry.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZPTCH HELP
  ZPTCH ?
- Only commands entered from the terminal where the ZPTCH BEGIN command was entered are included in the patch deck. Commands entered from other terminals are not included in the patch deck.
- The LIST parameter lists all the changes included in a patch deck before the patch deck is run. Therefore, if the patch deck contains more than one change for the same storage location, the information displayed for each change does not reflect the previous change. See the examples for more information.
- All verification requests are performed before any data is changed. If any verification in the patch deck fails, the patch deck is not run.

Examples

The ABC patch deck is built in the following example.

```
User:   ZPTCH BEGIN NAME-ABC

System: PTCH0003I 10.48.54 QUEUEING FOR PATCH DECK ABC STARTED

User:   ZACOR B2C 2F VAL-1F071C92

System: PTCH0001I 10.48.54 ITEM ADDED TO PATCH DECK ABC

User:   ZADCA WGT 8 00000000

System: PTCH0001I 10.48.54 ITEM ADDED TO PATCH DECK ABC

User:   ZAPGM CC3705 30 B7884340 COPY-C

System: PTCH0001I 10.48.54 ITEM ADDED TO PATCH DECK ABC

User:   ZPTCH STOP

System: PTCH0005I 10.48.54 QUEUEING FOR PATCH DECK ABC STOPPED
```

The JERRYG patch deck is displayed in the following example.

```
User:   ZPTCH DISP NAME-JERRYG

System: PTCH0006I 13.13.50 DISPLAY OF PATCH DECK JERRYG
         1 ZAPGM CVZZ   00000148 4700
                        VERSION-      LOADSET- BASE
         2 ZAPGM CCSNA1 0000118A 9580
                        VERSION-      LOADSET-
         3 ZAPGM CHUG   00000148 4700
                        VERSION- JW   LOADSET- GRATEFU
         4 ZACOR        00CE1980 0000
                        IS- 01
         5 ZADCA SNA    00000408 0861
                        IS- 01
        END OF DISPLAY
```

The ABC patch deck is run in the following example. Each change is displayed as it is made.

```
User:   ZPTCH EXECUTE NAME-ABC LIST

System: PTCH0009I 10.48.54 EXECUTION STARTING FOR PATCH DECK ABC
        ACOR0010I 10.48.54 BEGIN DISPLAY
         00000B2C- 1F071C92 EFD20000 00000000 00000000 .....K.. ........
         00000B3C- 000077AC 00087800 0004F600 0004F6BA ........ ..6...6.
        ALTERED TO-
         00000B2C- 2F071C92 EFD20000 00000000 00000000 .....K.. ........
         00000B3C- 000077AC 00087800 0004F600 0004F6BA ........ ..6...6.
        ADCA0010I 10.48.54 BEGIN DISPLAY
         011AC460- 3200001F 00000000 15766102 C0180000 ........ ../.....
         011AC470- 00000001 00003D40 000000C2 3200000F ........ ...B....
        ALTERED TO-
         011AC460- 00000000 00000000 15766102 C0180000 ........ ../.....
         011AC470- 00000001 00003D40 000000C2 3200000F ........ ...B....
        APGM0010I 10.48.54 BEGIN DISPLAY
         00000030- B7884340 70004100 00024144 90784120 ... .... ........
         00000040- A0084830 40064930 20004780 A08C4120 .... ... ........
        ALTERED TO-
         00000030- B7884340 70004100 00024144 90784120 ... .... ........
         00000040- A0084830 40064930 20004780 A08C4120 .... ... ........
                         EXECUTION COMPLETE FOR PATCH DECK ABC
```

The first entry in the ABC patch deck is removed in the following example.

```
User:   ZPTCH REMOVE NAME-ABC ENTRY-1

System: PTCH0008I 10.48.54 PATCH DECK ENTRY REMOVED
```

The ABC patch deck is deleted in the following example.

```
User:   ZPTCH PURGE NAME-ABC

System: PTCH0002I 10.48.54 PATCH DECK ABC PURGED
```

In the following example, a patch deck containing 2 change requests for the same storage address is built. Notice that when the patch deck is run, the LIST parameter lists each change in the patch deck before the change actually takes place.

```
User:   ZPTCH BEGIN NAME-TEST

System: PTCH0003I 09.35.19 QUEUEING FOR PATCH DECK TEST STARTED

User:   ZACOR 100000 AAAAAAAABBBBBBBBCCCCCCCCDDDDDDDD

System: PTCH0001I 09.35.42 ITEM ADDED TO PATCH DECK TEST

User:   ZACOR 100000 EEEEEEEEFFFFFFFF

System: PTCH0001I 09.36.00 ITEM ADDED TO PATCH DECK TEST

User:   ZPTCH STOP

System: PTCH0005I 09.36.08 QUEUEING FOR PATCH DECK TEST STOPPED

User:   ZPTCH EXECUTE NAME-TEST LIST

System: PTCH0009I 09.37.23 EXECUTION STARTING FOR PATCH DECK TEST
        ACOR0010I 09.37.23 BEGIN DISPLAY
         00100000- 40404040 40404040 40404040 40404040
         00100010- 40404040 40404040 40404040 40404040
        ALTERED TO-
         00100000- AAAAAAAA BBBBBBBB CCCCCCCC DDDDDDDD ........ ........
         00100010- 40404040 40404040 40404040 40404040
        ACOR0010I 09.37.23 BEGIN DISPLAY
         00100000- 40404040 40404040 40404040 40404040
         00100010- 40404040 40404040 40404040 40404040
        ALTERED TO-
         00100000- EEEEEEEE FFFFFFFF 40404040 40404040 ........
         00100010- 40404040 40404040 40404040 40404040
                            EXECUTION COMPLETE FOR PATCH DECK TEST

User:   ZDCOR 100000.20

System: DCOR0010I 09.39.15 BEGIN DISPLAY
         00100000- EEEEEEEE FFFFFFFF CCCCCCCC DDDDDDDD ........ ........
         00100010- 40404040 40404040 40404040 40404040
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

Related Information

See *TPF Main Supervisor Reference* for more information about patch decks.

ZPUMP–Police Unsolicited Message Directory Items

Use this command to delete any redundant or unsolicited messages and to notify agents that have undisplayed unsolicited messages.

Requirements and Restrictions

The TPF system must be in CRAS state or higher.

Format

```
►►──ZPUMP─────────────────────────────────────────────────────────────►◄
```

Additional Information

This command is a time-initiated command that is issued automatically by the TPF system each hour. You can also enter it at any time.

Examples

Redundant, unsolicited messages that were not yet displayed are deleted in the following example.

```
User:   ZPUMP

System: SMPB0001I 12.20.58 REQUEST PROCESSED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the unsolicited message processor.

ZRBKD–Recoup Descriptor (BKD) Functions

Use this command to:

- Display recoup descriptors (BKDs) by name or slot number
- Move recoup descriptors
- Create dummy recoup descriptors
- Restore recoup descriptors
- Initialize the BKD load control record (BK0LC)
- Display recoup descriptor update history.

Note: Enter the ZOLDR LOAD command to load recoup descriptors to the TPF system.

Requirements and Restrictions

- You cannot enter this command while recoup processing is active.
- You can enter the following only from the prime CRAS:
  - ZRBKD DISP ALL
  - ZRBKD MOVE *nameversion*
  - ZRBKD REST N*slotnum*
  - ZRBKD INIT CTL
  - ZRBKD DUMM N*slotnum*
- You can enter the following only in test mode or from the prime CRAS:
  - ZRBKD MOVE ALL
  - ZRBKD DUMM ALL
  - ZRBKD REST ALL

Format

```
►►──ZRBKD── ──┬─DISP── ──┬─ALL────────┬────────────────────┬──────────►◄
              │          ├─name───────┤                    │
              │          ├─Aslotnum───┤                    │
              │          └─Nslotnum───┘                    │
              ├─MOVE── ──┬─ALL───────────────────────────┬─┤
              │          └─nameversion──┬───────────────┬┘ │
              │                         └─ ──Nslotnum───┘  │
              ├─DUMM── ──┬─ALL────────┬────────────────────┤
              │          └─Nslotnum───┘                    │
              ├─REST── ──┬─ALL────────┬────────────────────┤
              │          └─Nslotnum───┘                    │
              ├─INIT CTL───────────────────────────────────┤
              └─HIST───────────────────────────────────────┘
```

DISP
displays recoup descriptors where:

ALL
displays all active recoup descriptors.

name
is the 4-character name of a specific recoup descriptor that you want to display.

Note: Recoup descriptor names must start with the letters BK.

A*slotnum*
displays all active recoup descriptors starting from a specific slot number, where *slotnum* is the 2-digit slot number from 00 to 99.

N*slotnum*
displays all active recoup descriptors, including dummy descriptors, starting from a specific slot number, where *slotnum* is the 2-digit slot number from 00 to 99.

MOVE
moves active recoup descriptors, where:

ALL
moves all active recoup descriptors into current slot numbers. Descriptors that were in current slot numbers are moved to previous slot numbers.

name
is a specific 4-character recoup descriptor name.

Notes:
1. Recoup descriptor names must start with the letters BK.
2. If you do not specify N*slotnum*, the recoup descriptor is moved into its current slot number.

version
is a specific version of the named recoup descriptor.

N*slotnum*
moves the specified recoup descriptor into a specific slot number, where *slotnum* is the 2-digit slot number from 00 to 99.

DUMM
creates dummy recoup descriptors, where:

ALL
creates dummy recoup descriptors and puts them into all the slot numbers from 0 to 99.

N*slotnum*
creates a dummy recoup descriptor and puts it into a specific slot number, where *slotnum* is the 2-digit slot number from 00 to 99.

Note: The recoup dummy chain chase descriptor segment (BKDW) must be loaded to use the DUMM parameter.

REST ALL
restores recoup descriptors, where:

ALL
restores all recoup descriptors to the original sequence numbers.

N*slotnum*
restores the specified recoup descriptor into its original sequence number, where *slotnum* is a 2-digit slot number from 00 to 99.

INIT CTL
initializes the BKD load control record (BK0LC).

HIST
displays the history of the last 10 commands that changed recoup descriptors.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZRBKD HELP
ZRBKD ?

Examples

Recoup information for all active descriptors is displayed in the following example, where:

NO
indicates the sequence number that the BKD occupies.

STAT
indicates one of the following descriptor statuses:

D indicates the BKD is a dummy.

M indicates the BKD was moved.

R indicates the BKD was restored.

S indicates the BKD was saved.

CURR
indicates the current descriptor occupying the sequence number.

PREV
indicates the descriptor that is used to occupy the sequence number.

GR
indicates the number of primary groups in the descriptor.

ID indicates the record IDs to chase for the descriptor.

VER
indicates a specific version of the named recoup descriptor.

```
User:   ZRBKD DISP ALL

System: BRUB0012I 10.01.31 RECOUP DESCRIPTOR TABLE
        NO STAT  CURR   PREV     GR ID  -VER
        03 .MS. BKZHP1 BKZHP1    01 JO  -000
        04 .MS. BKZIP1 BKZIP1    02 JS  -001 JS  -002
        05 .MS. BKZJP1 BKZJP1    01 JH  -000
        06 .MS. BKZ6P1 BKZ6P1    01 JA  -000
        07 .MS. BKZMP1 BKZMP1    01 JG  -000
        10 .MS. BKZKP1 BKZKP1    01 JT  -000
        11 .MS. BKZLP1 BKZLP1    02 JW  -000 JW  -006
        12 .MS. BKZCP1 BKZCP1    02 JW  -050 JA  -051
        17 .MS. BKD740 BKDW40    01 FC2A-000
        18 .MS. BKZTP1 BKZTP1    01 FC13-000
        20 .MS. BKDY40 BKDW40    04 FC33-000 FC32-000 FC35-000
                                           CD  -000
        END OF DISPLAY
```

The following example moves descriptor BKZ4P1 into slot number 4.

```
User:   ZRBKD MOVE BKZ4P1 N04

System: BRUC0006I 15.34.34 REQUEST PROCESSED
```

The following example displays the history of the last 10 commands that changed recoup descriptors (BKDs).

```
User:   ZRBKD HIST

System: BRUB0013I 10.00.59 RECOUP DESCRIPTOR TABLE UPDATE HISTORY
          DATE       TIME     ENTRY
        12DEC2000  10.46.34   ZRBKD DUMM N13
        06DEC2000  10.33.25   ZRBKD MOVE BKDY40 N20
        06DEC2000  10.33.19   ZRBKD MOVE BKZTP1 N18
        06DEC2000  10.33.13   ZRBKD MOVE BKD740 N13
        06DEC2000  10.33.09   ZRBKD MOVE BKZ6P1 N06
        06DEC2000  10.33.04   ZRBKD MOVE BKZNP1 N13
        06DEC2000  10.33.00   ZRBKD MOVE BKZMP1 N07
        06DEC2000  10.32.54   ZRBKD MOVE BKZLP1 N11
        06DEC2000  10.32.49   ZRBKD MOVE BKZKP1 N10
        06DEC2000  10.32.45   ZRBKD MOVE BKZJP1 N05
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRCBI–Initialize Routing Control Blocks

Use this command to initialize routing control blocks (RCBs) for all systems in the network.

Requirements and Restrictions

The TPF system must be in 1052 state or higher.

Format

```
►►──ZRCBI── ──┬─ALL── ──cpuid──────────────────────┬───────────────────►◄
              ├─MOD── ──number─────────────────────┤
              └─SGL── ──lniata── ──cpuid──┬───────┬┘
                                          └─ ──city─┘
```

ALL *cpuid*
: initializes the RCBs for the specified processor, where *cpuid* is the 1-character CPU ID of a processor.

MOD *number*
: initializes the RCBs for the specified module, where *number* is the 3-digit hexadecimal number of a module.

SGL
: initializes the RCB for the specified console.

lniata
: is the 6-digit hexadecimal line number, interchange address, and terminal address of a console.

cpuid
: is the 1-character alphanumeric CPU ID of a processor.

city
: is the city code for a console.

Additional Information

None.

Examples

The RCB for the specified console is initialized in the following example.

```
User:   ZRCBI SGL 4E0000B

System: RCBI0005I 12.40.02 RCB INITIALIZATION COMPLETE
```

The RCBs for processor B are initialized in the following example.

```
User:   ZRCBI ALL B

System: RCBI0000I 12.41.29 ALL PROCESSED
        NUMBER INITIALIZED     303
        INVAL TERM ADDRS         0
        INVAL TERM TYPES         0
        NUMBER ATTEMPTED       303
```

Related Information

See *TPF Data Communications Services Reference* for more information about the message routing package.

ZRCRS–Route Message to CRAS

Use this command to send a message to the computer room agent set (CRAS) from a high-speed terminal or a display device.

Requirements and Restrictions

None.

Format

```
►►──ZRCRS── ──message──────────────────────────────────────────────►◄
```

message
is 1–252 character message, which includes end-of-message (EOM) characters and carrier return characters.

Additional Information

None.

Examples

In the following example, a message is sent to the CRAS terminal.

```
User:   ZRCRS NETWORK TRAFFIC STARTED

System: RCRS0003I 14.26.38 MESSAGE SENT TO PRIME CRAS FOR PROCESSOR B
```

Related Information

See *TPF Non-SNA Data Communications Reference* for more information about CRAS support.

ZRDIR CAPTURE–Capture Pool Directories and Keypoint 9

Use this command to capture the pool rollin directory (#SONRI) and keypoint 9 (CTK9) so they can be used for a potential restore at a later time. This command copies the pool rollin directory (#SONRI) to the recoup SONRI save area (#SONSV), keypoint 9 to an #IBMM4 record, and the fixed file copies of the pool keypoint tables (CY2KTs) to the CY2KT save area (#CY2CPY).

Requirements and Restrictions

- You cannot enter this command while recoup processing is active.
- If you have a loosely coupled configuration with processors IPLed from TPF system images with pool expansion (PXP) support and processors IPLed from images with 32-way loosely coupled pool support, you must keep track of the pool support on the processor from which this command was entered. You must enter the ZRDIR START RESTORE command from a processor IPLed with the same pool support.

  Enter the ZPMIG STATUS command to determine the format of the pool data structures on the processor.

Format

```
►►──ZRDIR CAPTURE──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example captures the #SONRI, CTK9, and fixed file copies of the CY2KTs so they can be used for a potential restore at a later time.

```
User:    ZRDIR CAPTURE

System:  BCPY0001I 15.23.43  - KEYPOINT 9 AND SONRI SAVE STARTED
         BCPY0002I 15.23.43  - KEYPOINT 9 AND SONRI SAVE COMPLETE
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRDIR START RESTORE–Restore Pool Directories and Keypoint 9

Use this command after a recoup phase 3 rollin to restore the pool rollin directory (#SONRI), keypoint 9 (CTK9), and the fixed file copies of the pool keypoint tables (CY2KTs) if there are problems with the rollin. This command copies the recoup SONRI save area (#SONSV) to #SONRI, the saved keypoint 9 in the #IBMM4 record to the working copy of keypoint 9, and the saved copies of the pool keypoint tables (#CY2CPY) to the working CY2KTs in the #CY2KT fixed file record type.

Note: All pseudo directories that were in main storage between the capture and the restore are zeroed, and file pools are reconciled by updating keypoint 9 and getting the correct file pool counts.

Requirements and Restrictions

- You can enter this command only if the TPF system is in 1052 state.
- You cannot enter this command while recoup processing is active.
- If you have a loosely coupled configuration with processors IPLed from TPF system images with pool expansion (PXP) support and processors IPLed from images with 32-way loosely coupled pool support, you must enter this command from a processor that was IPLed with the same pool support as the processor from which the ZRDIR CAPTURE command was entered.

  Enter the ZPMIG STATUS command to determine the format of the pool data structures on the processor.

Format

```
►►──ZRDIR START RESTORE──────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example restores the #SONRI, CTK9, and fixed file copies of the CY2KTs after a problem occurred with the recoup phase 3 rollin.

```
User:   ZRDIR START RESTORE

System: START DIRECTORY RESTORE
        DIRECTORY RESTORE COMPLETE
        RFPC0001I 14.39.03  - STARTED RECONCILIATION OF SON POOL COUNTS
        RFPC0002I 14.39.03  - END OF JOB OF SON POOL RECONCILIATION
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRECP ABORT–Abort Recoup

Use this command to end recoup phase processing.

Requirements and Restrictions

- You can enter this command only when recoup processing is active or between phases.
- You cannot enter this command when the NCB reorganization function is running.

  Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►──ZRECP ABORT──┬──────┬──────────────────────────────────────────────►◄
                 └──BP──┘
```

BP
: resets recoup indicators that need to be reset when a hard error occurs (for example, a 000004 system error).

Additional Information

None.

Examples

The following example ends recoup processing.

```
User:   ZRECP ABORT

System: SNAP0001I 10.40.18 CPU-B SS-BSS SSU-HPN IS-1
        PSW-071D0000 847A810A PGM-BRPE CODE-I00041001 TERM-010000B
        RECOUP IS ABORTING
        END OF SNAPC CONSOLE DISPLAY
        RECP004EI 10.40.21 RECOUP ABORT IN PROGRESS
        RECP00FAI 10.40.27 PSEUDO DIRECTORY FLUSH STARTED
        RECP00FCI 10.40.27 PSEUDO DIRECTORY FLUSH COMPLETE
        RECP001AT 10.40.27 RECOUP ABORTED
```

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.
- See *TPF ACF/SNA Data Communications Reference* for more information about the node control block (NCB) reorganization function.

ZRECP ADD–Add a Record ID to the Rebuild Tables

Use this command to add record IDs to the exclusion or inclusion table before entering the ZRECP REBUILD command.

The *exclusion table* is a list of record IDs that are to be excluded from the second recoup rollin; in other words, all lost addresses with these record IDs are not rolled in, but protected.

The *inclusion table* is a list of record IDs that are to be rolled in during the second recoup rollin; in other words, all lost addresses with these record IDs are rolled in. All lost addresses with other record IDs are not rolled in, but protected.

The exclusion and inclusion tables cannot be used together. Only one of the two can be valid for a particular recoup run.

Requirements and Restrictions

None.

Format

```
                               ┌─  ──┐
                               ▼     │
►►──ZRECP ADD── ──┬─IN─┬── ───recid──┴──────────────────────────────►◄
                  └─EX─┘
```

IN adds the specified record ID or IDs to the recoup inclusion table.

EX
adds the specified record ID or IDs to the recoup exclusion table.

recid
is a 2-digit character or 4-digit hexadecimal ID of a record type. You can specify as many as 10 record IDs at a time. The inclusion or exclusion table can contain a maximum of 50 record IDs.

Additional Information

None.

Examples

The following example adds record IDs CA, FC11, CD, and FC18 to the inclusion table.

```
User:   ZRECP ADD IN CA FC11 CD FC18

System: BRB20010I 10.54.23 ID SET COMPLETE - ENTER ZRECP DISPLAY INC TO VERIFY
```

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.

- See *TPF ACF/SNA Data Communications Reference* for more information about the node control block (NCB) reorganization function.

ZRECP CONTINUE–Continue Recoup Phase 1

Use this command to complete recoup phase 1 processing after receiving fixed errors on one or more than one record. Records with fixed errors are not protected by recoup processing and can result in a database integrity loss. Therefore, use this command carefully.

Instead of entering this command, consider doing the following:

1. Correct all record IDs that received fixed errors during recoup chain chase processing.
2. Selectively add those record IDs to the recoup run.

Requirements and Restrictions

- You can enter this command only in response to message RECP0014A.
- You cannot enter this command when the node control block (NCB) reorganization function is running.

  Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.
- All outstanding ZRECP SEL command requests must be completed before you enter the ZRECP CONTINUE command, otherwise a sequence error message (RECP001CW) will be displayed.

Format

```
►►──ZRECP CONTINUE────────────────────────────────────────────────►◄
```

Additional Information

- For TPF recoup processing, all fixed errors are displayed on the console.
- If you have the TPFDF product installed, you can enter the ZRECP ONEL command to display fixed errors online.
- For TPFDF recoup processing, if the main monitor entry control block (ECB) times out, fixed errors will not be displayed on the console or in the online error log. If this occurs, run recoup again or exclude the entire TPFDF database before you enter ZRECP PROCEED to roll in the records.

Examples

The following example completes recoup phase 1 processing without correcting records with fixed errors.

```
System: RECP0012A 09.50.21 FIXED ERROR-RESPOND
        RECP0014A 09.50.21 DEFERING TIL SEL RECOUP COMPLETE OR CONTINUE RESPOND

User:   ZRECP CONTINUE

System: RECP00F6I 09.50.32 PSEUDO DIRECTORY VERIFY STARTED+
        CSMP0097I 09.50.32 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP00F8I 09.50.32 PSEUDO DIRECTORY VERIFY COMPLETE+
        CSMP0097I 09.50.32 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP00FAI 09.50.32 PSEUDO DIRECTORY FLUSH STARTED+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP00FCI 09.50.33 PSEUDO DIRECTORY FLUSH COMPLETE+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        COTC0300A 09.50.33 TCLS BSS    REMOVE RCP FROM DEVICE 425
                           VSN A00289 G0001 S0001 F38K     SL  BLK    NOCOMP +
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP00FEI 09.50.33 PHASE 2 START+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP0016I 09.50.33 PHASE I COMPLETED+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP0016I 09.50.33 PHASE I COMPLETED+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF40010I 09.50.33 RECOUP INTEGRITY CHECKING STARTED_
        RECP00F6I 09.50.33 PSEUDO DIRECTORY VERIFY STARTED+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        RECP00F8I 09.50.33 PSEUDO DIRECTORY VERIFY COMPLETE+
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF40011I 09.50.33 RECOUP INTEGRITY CHECKING COMPLETE_
        CSMP0097I 09.50.33 CPU-A SS-BSS  SSU-BSS  IS-01
        BRID0003I 09.50.33 BEGIN ID TABLE MERGE+
        CSMP0097I 09.50.34 CPU-A SS-BSS  SSU-BSS  IS-01
        BRID0004I 09.50.34 END ID TABLE MERGE+
        CSMP0097I 09.50.34 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF90001I 09.50.34 PHASE 2 COMPLETED+
        CSMP0097I 09.50.34 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF20014I 09.50.34 RECOUP - POOLS IN  USE  SUMMARY DISPLAY

        POOL                         TOTALS                  IN  USE
        -------- ------------------------- -------------------------
        SST BASE                     28 545                        0 _
        SDP BASE                    280 005                      102
        LST BASE                     17 226                        0
        LDP BASE                    282 546                       23
        4ST BASE                      5 400                        0
        4DP BASE                    153 128                       10 _
        4D6 BASE                     82 008                        0
        END DISPLAY+
        CSMP0097I 09.50.34 CPU-A SS-BSS  SSU-BSS  IS-01
        BOFI0026I 09.50.34 RECOUP PHASE 1 ID COUNTS DISPLAY
          ID                        COUNTS
        ------- -------------------------
        D1C8 JH                          2
        D1C9 JI                          5
        D1D1 JJ                          6 _
        D1D2 JK                          6
        D1D3 JL                          8
        D1D4 JM                         11
        D1D5 JN                         97
        END OF ID TABLE DISPLAY+
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP DEL–Delete a Record ID from the Rebuild Tables

Use this command to delete record IDs from the exclusion or inclusion table before entering the ZRECP REBUILD command.

The *exclusion table* is a list of record IDs that are to be excluded from the second recoup rollin; in other words, all lost addresses with these record IDs are not rolled in, but protected.

The *inclusion table* is a list of record IDs that are to be rolled in during the second recoup rollin; in other words, all lost addresses with these record IDs are rolled in. All lost addresses with other record IDs are not rolled in, but protected.

The exclusion and inclusion tables cannot be used together. Only one of the two can be valid for a particular recoup run.

Requirements and Restrictions

None.

Format

```
                                ┌─  ───┐
                                ▼      │
►►──ZRECP DEL── ──┬─IN─┬── ───recid────┴──────────────────────────────►◄
                  └─EX─┘
```

IN deletes the specified record ID or IDs from the recoup inclusion table.

EX
deletes the specified record ID or IDs from the recoup exclusion table.

recid
is a 2-digit character or 4-digit hexadecimal ID of a record type. You can specify as many as 10 record IDs at a time. The inclusion or exclusion table can contain a maximum of 50 record IDs.

Additional Information

None.

Examples

The following example deletes the CD record ID from the inclusion table.

```
User:   ZRECP DEL IN CD

System: BRB20010I 11.43.03 ID SET COMPLETE - ENTER ZRECP DISPLAY INC TO VERIFY
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP DISPLAY–Recoup Display Options

Use this command to display information about different phases of recoup.

Requirements and Restrictions

You can enter this command only after recoup phase 1 processing is completed.

Format

```
                  ┌─ ──COunts──────────────────────────────────────────┐
►►──ZRECP DISPLAY─┼────────────────────────────────────────────────────┼──────►◄
                  ├─ ──EXclude─────────────────────────────────────────┤
                  ├─ ──INclude─────────────────────────────────────────┤
                  ├─ ──FARF────────────────────────────────────────────┤
                  │            ┌─ ──Section-SUM──────────┐             │
                  ├─ ──POOLs───┼─────────────────────────┼─────────────┤
                  │            ├─ ──Section-ALL──────────┤             │
                  │            └─ ──Section-pooldevtype──┘             │
                  │     ┌─ ───────────────┐                            │
                  │     ▼                 │                            │
                  ├─ ───────┬─ID-recid────┴────────────────────────────┤
                  │         ├─PROC-cpuid─┤                             │
                  │         └─SSU-ssuid──┘                             │
                  ├─ ──DEAFA── ──┬─TYPE-pooldevtype─┬── ──SKIP-nn──────┤
                  │              └─DIR-ordnum───────┘                  │
                  │             ┌─ ──REC-SUM──┐                        │
                  ├─ ──VERify───┼─────────────┼────────────────────────┤
                  │             ├─ ──REC-ALL──┤                        │
                  │             └─ ──ID-recid─┘                        │
                  ├─ ──LOst─────┬──────────────┬───────────────────────┤
                  │             └─ ──ID-recid──┘                       │
                  └─ ──ERron─────┬──────────────┬──────────────────────┘
                                 └─ ──ID-recid──┘
```

COUNTS
displays recoup phase 1 counts for all SSU record IDs.

EXCLUDE
displays the list of record IDs in the recoup exclusion table. (The recoup exclusion table is used by the ZRECP REBUILD command to protect record IDs from becoming available to the TPF system).

INCLUDE
displays the list of record IDs in the recoup inclusion table. (The recoup inclusion table is used by the ZRECP REBUILD command to make record IDs available to the TPF system).

FARF
displays a summary of file address references.

Note: You can specify this parameter only if you have the TPFDF product installed and recoup runtime options were set to log FARF references.

POOLS
displays recoup phase 2 merged pseudo directory counts and releases that were found after ZRPDU CREATE command processing.

SECTION
displays how pool counts will be displayed, where:

SUM
displays a summary of all counts by pool section.

ALL
displays counts by pool segments for every pool section.

pool
is one of the following pool types:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

devtype
is a device type of A, B, C, or D.

ID displays phase 1 counts for the specified record ID.

recid
is a 2-character EBCDIC or 4-character hexadecimal ID of a record type.

PROC
specifies a processor, where:

cpuid
is the 1-character ID of the processor for which you want to display phase 1 counts.

SSU
displays phase 1 counts for the specified subsystem user (SSU). If you do not specify this parameter, all SSU counts are displayed.

ssuid
is the ID of an SSU.

DEAFA
displays the in-use addresses in a deactivated pool section. Use this command to display as many as 200 records.

TYPE
specifies the pool and device type addresses that you want to display.

DIR-*ordnum*
displays a specific directory ordinal number, where *ordnum* is the directory ordinal number specified.

SKIP-*nn*
specifies the number of addresses to skip before displaying, where *nn* is a decimal number.

VERIFY
displays all pool records that were found more than once by recoup phase 1.

Note: You can specify this parameter only if you have a loosely-coupled TPF system and the TPFDF product is installed.

REC
displays each duplicate record ID that was found, where:

ALL
display details about each record ID.

SUM
displays a summary of all record IDs.

LOST
displays the counts of lost pool files by each record ID. This parameter is valid only if the TPF system is in NORM state and the lost addresses pass of recoup phase 3 has completed.

ERRON
displays the counts of erroneously available pool files by each record ID. This parameter is valid only if the TPF system is in NORM state and the erroneously available pass of recoup phase 3 has completed.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZRECP DISPLAY HELP
ZRECP DISPLAY ?

Examples

The following example displays the total recoup phase 1 counts for all SSU record IDs.

```
User:   ZRECP DISPLAY

System: BOFI0026I 12.11.41 RECOUP PHASE 1 ID COUNTS DISPLAY
          ID                        COUNTS
        ------- --------------------------
        C3C1 CA                         50
        D1C2 JB                         10
        D1C3 JC                          4
        D1C4 JD                          4
        D1C5 JE                          6
        D1C6 JF                         22
        D1C8 JH                          4
        D1C9 JI                         10
        D1D1 JJ                         12
        D1D2 JK                         12
        D1D3 JL                         16
        D1D4 JM                         22
        D1D5 JN                        194
        D1D6 JO                         12
        D1D7 JP                        294
        D1D9 JR                       1920
        D1E4 JU                         24
        D1E5 JV                          6
        D1E6 JW                          4
        D1E7 JX                          6
        D1E8 JY                         31
        D1D5 JN                        194
        D1D6 JO                         12
        D1D7 JP                        294
        D1D9 JR                       1920
        D1E4 JU                         24
        D1E5 JV                          6
        D1E6 JW                          4
        D1E7 JX                          6
        D1E8 JY                         31
        D1F1 J1                          2
        D1F2 J2                       3892
        D1F3 J3                          1
        D1F4 J4                         13
        D1F5 J5                         27
        D1F6 J6                         46
        D1F7 J7                          2
        D1F8 J8                          3
        D1F9 J9                          3
        D2C1 KA                          1
        D2C2 KB                          1
        D2C3 KC                          3
        D2C4 KD                          3
        D2C6 KF                          3
        D2F8 K8                         14
        FC10                           208
        FC11                             4
        FC15                            21
        D2C1 KA                          1
        D2C2 KB                          1
        D2C3 KC                          3
        D2C4 KD                          3
        D2C6 KF                          3
        D2F8 K8                         14
        FC10                           208
        FC11                             4
        FC15                            21
        FC16                           118
        FC33                             1
        END OF ID TABLE DISPLAY
```

The following example displays the list of record IDs in the recoup inclusion table.

```
User:   ZRECP DISPLAY INC

System: BRB30001I 10.35.44 INCLUSION ID TABLE
                  JA/D1C1  ../....  ../....  ../....  ../....
```

The following example displays the in-use addresses in a deactivated pool section for small duplicate pool (SDP) addresses.

```
User:   ZRECP DISPLAY DEAFA TYPE-LDPB SKIP-40000

System: B1A80004I 11.41.37 DISPLAY OF INUSE ADDRESSES OF DEACTIVATED SECTIONS
        -------------------  -------------------  -------------------
        0C/0000000000036844  0C/0000000000036845  0C/0000000000036846 _
        0C/0000000000036847  0C/0000000000036848  0C/0000000000036849
        0C/000000000003684A  0C/000000000003684B  0C/000000000003684C
        0C/000000000003684D  0C/000000000003684E  0C/000000000003684F
        0C/0000000000036850  0C/0000000000036851  0C/0000000000036852
        0C/0000000000036853  0C/0000000000036854  0C/0000000000036855 _
        0C/0000000000036856  0C/0000000000036857  0C/0000000000036858
        0C/0000000000036859  0C/000000000003685A  0C/000000000003685B
        0C/000000000003685C  0C/000000000003685D  0C/000000000003685E
        0C/000000000003685F  0C/0000000000036860  0C/0000000000036861
                                        .
                                        .
                                        .
        0C/00000000000368F8  0C/00000000000368F9  0C/00000000000368FA _
        0C/00000000000368FB  0C/00000000000368FC  0C/00000000000368FD
        0C/00000000000368FE  0C/00000000000368FF  0C/0000000000036900 _
        0C/0000000000036901  0C/0000000000036902  0C/0000000000036903
        0C/0000000000036904  0C/0000000000036905  0C/0000000000036906
        0C/0000000000036907  0C/0000000000036908  0C/0000000000036909
        B1A80015I 11.43.52 END OF DISPLAY
```

The following example displays the counts of lost pool files by each record ID.

```
User:   ZRECP DISPLAY LOST

System: BOFI0028I 12.17.30 RECOUP LOST ADDRESS ID TABLE DISPLAY
          ID                      COUNTS
        ------- -------------------------
        C4D5 DN                        2
        C3D3 CL                      715
        C4F7 D7                        3
        C4F8 D8                        1
        C5F1 E1                        2
        C5F2 E2                        1
        C5F7 E7                        1
        C5F8 E8                       62
        C8C1 HA                        1
        C8D9 HR                        1
        C9C4 ID                     1185
        C9C6 IF                    11143
        C9D4 IM                     3536
        CB19                        1946
        D00D                           3
        D3C1 LA                     1379
        D3D6 LO                   105607
        D3E3 LT                      989
        D4C5 ME                        2
        D4C6 MF                        3
        D4D4 MM                    17299
        D4D8 MQ                      910
        F6C6 6F                       28
        F6C7 6G                      224
        FA02                          45
        FC33                     1349750
        END OF ID TABLE DISPLAY
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP DUMP–Start or Stop Recoup Phase 5

Use this command to start or stop recoup phase 5.

Requirements and Restrictions

You cannot enter this command when the NCB reorganization function is running.

Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►──ZRECP DUMP──┬────────────┬──────────────────────────────────────────►◄
                ├──RESTART───┤
                └──ABORT─────┘
```

RESTART
starts recoup phase 5 after you perform a system restart.

ABORT
stops recoup phase 5.

Additional Information

Enter **ZRECP DUMP** (without any parameters) to start recoup phase 5.

Examples

Recoup phase 5 is started in the following example.

```
User:   ZRECP DUMP

System: RECP0000I 15.23.53 RECP SELD COMPLETED
        COTC0087A 15.23.53 TCLS BSS1   REMOVE ADR FROM DEVICE 42B
                   VSN A00142  NOBLK
```

Recoup phase 5 is stopped in the following example.

```
User:   ZRECP DUMP ABORT

System: RECP0000T 10.21.39 RECP SELD ABORTED
```

Related Information

- See *TPF Database Reference* for more information about online file recoup.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP ELOG–Turn Error Log On or Off

Use this command to turn on or turn off the online error logging.

Requirements and Restrictions

You can enter this command only if you have the TPFDF product installed.

Format

```
►►──ZRECP ELOG── ──┬─ON──┬──────────────────────────────────────────►◄
                   └─OFF─┘
```

ON
turns on error logging.

OFF
turns off error logging.

Additional Information

None.

Examples

The following example turns on error logging.

```
User:   ZRECP ELOG ON

System: BRCP0007I 13.47.39 ONEL FUNCTION COMPLETE+
```

The following example turns off error logging.

```
User:   ZRECP ELOG OFF

System: BRCP0007I 13.47.39 ONEL FUNCTION COMPLETE+
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP EXIT–End and Nullify Recoup in a Secondary Processor

Use this command to end and nullify recoup in a secondary processor.

If you enter the ZRECP EXIT command on a processor while that processor is chain chasing TPF recoup structures, all that chain chase work is nullified and redone on other processors.

If you enter the ZRECP EXIT command on a processor while that processor is chain chasing TPFDF recoup structures, all that chain chase work is nullified and redone on the other processors, but all previous TPF recoup chain chase work is preserved.

The ZRECP EXIT command is used for the following conditions.

- The target processor was hard-IPLed and virtual file access (VFA) was cleared.
- There are problems such as memory corruption and system outages, where the status of the processor is questionable.

When these conditions occur, determine whether to continue the recoup run or force recoup processing to end and start again.

Requirements and Restrictions

Before entering the ZRECP EXIT command, ensure that the target processor is in 1052 state or above.

The ZRECP EXIT command is not allowed if the target processor is still performing chain chasing activity. For this, enter the ZRECP STOP command first and, once that is completed, enter the ZRECP EXIT command. You do not have to enter the ZRECP STOP command if the target processor has just been IPLed and there are no recoup entry control blocks (ECBs) active on that processor.

Format

```
►►──ZRECP── ──PROC── ──cpuid── ──EXIT──────────────────────────────────────►◄
```

PROC *cpuid*
: specifies a processor, where *cpuid* is the 1-character ID of the processor for which you want to end and nullify recoup.

Additional Information

- Pseudo directories for the exited processor are still merged with pseudo directories of the other processors during recoup phase 2 processing. This ensures that in-use databases are protected. However, no integrity checking is done on these exited pseudo directories. They are merged regardless of their pool bit maps.
- Messages that are displayed in a secondary processor will also be displayed in the primary processor with the processor ID appended to the message to show where the message came from.

Examples

The following example ends and nullifies recoup processing on secondary processor C.

```
User:   ZRECP PROC C EXIT

System: RECP00C0I 18.09.35 "ZRECP PROC C EXIT" COMPLETE
        RECP00D0I 18.09.45 RECOUP RUN EXITED FROM THIS PROCESSOR  (PROC - C)"
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP FLUSH–Flush Pseudo Directory Records from VFA

Use this command to flush pseudo directory records from virtual file access (VFA) and to ensure that they were successfully filed out by reading them back in again.

Requirements and Restrictions

You can enter this command only when running under VM (VPARS).

Format

```
►►──ZRECP FLUSH──────────────────────────────────────────────────►◄
```

Additional Information

Records are automatically flushed from VFA at the end of recoup phase 1, recoup phase 3, and at the end of the pseudo directory build process during a pool directory update (PDU).

Examples

The following example flushes pseudo directory records from VFA and reads them back in.

```
User:   ZRECP FLUSH

System: RECP00FAI 10.40.27 PSEUDO DIRECTORY FLUSH STARTED
        RECP00FCI 10.40.27 PSEUDO DIRECTORY FLUSH COMPLETE
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP IGNORE–Bypass the First Recoup Rollin

Use this command to continue recoup processing when there are no erroneously available addresses to protect.

The total number of erroneously available addresses is shown when the ZRECP RESUME command is completed. If that number is 0, enter **ZRECP IGNORE** rather than ZRECP PROTECT. If the number is small, enter **ZRECP PROTECT**. If the number is very large, see your system programmer before responding with a command.

Requirements and Restrictions

- Not valid if running a selective recoup or if recoup was started with the TEST1 option.
- This command is valid only after ZRECP RESUME command processing has been completed successfully.

Format

```
►►──ZRECP IGNORE──┬──────────┬──────────────────────────────────►◄
                  └──EXCDUR──┘
```

EXCDUR
excludes pool addresses that were released during recoup processing from the final recoup rollin that could make these addresses available to the TPF system.

Note: Specify the EXCDUR parameter only if there was a problem with the release file storage (RFS) activity during recoup phase 1 processing that could cause the results of the RFS activity to be incorrect.

Additional Information

None.

Examples

The following example continues recoup processing without protecting any erroneously available addresses.

```
System: BCP00007I 13.46.38 RECOUP BUILD ERRONEOUS AVAILABLE STARTED
        BOFF0010I 13.46.38 PSEUDO DIR FLUSH STARTED
        BOFF0011I 13.46.38 PSEUDO DIR FLUSH COMPLETE
        RECP00F6I 13.46.38 PSEUDO DIRECTORY VERIFY STARTED
        RECP00F8I 13.46.38 PSEUDO DIRECTORY VERIFY COMPLETE
        BCP10016I 13.46.39 RECOUP 57 TOTAL ERRONEOUSLY AVAILABLE ADDRESSES
        BCP00008I 13.46.39 RECOUP BUILD ERRONEOUS AVAILABLE COMPLETE
        BCP00010A 13.46.39 RECOUP ENTER ZRECP PROTECT OR IGNORE TO CONTINUE

User:   ZRECP IGNORE

System: BCP00018I 13.48.58 RECOUP PROTECTION OF ERR0NEOUS AVAILABLE SKIPPED
        BCP00020I 13.48.58 RECOUP BACKUP OF SONUP STARTED
        BCP00021I 13.48.59 RECOUP BACKUP OF SONUP COMPLETE
        BCP00013I 13.48.59 RECOUP 'DURING' PHASE 1 RELEASE PASS STARTED
        BOFA0001I 13.48.59
               ONLINE PDU

        DOUBLE RELEASES   -        0
        FC33 RECS PROCESSED -      0
        CA RECS PROCESSED -        0
        STATUS  - I  (P=ACTIVE I=INACTIVE)
        BOFF0010I 13.48.59 PSEUDO DIR FLUSH STARTED
        BOFF0011I 13.48.59 PSEUDO DIR FLUSH COMPLETE
        BCP00014I 13.48.59 RECOUP 'DURING' PHASE 1 RELEASE PASS COMPLETE
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP LEVEL–Pause and Continue Recoup Phase 1

Use this command to pause recoup phase 1 or continue recoup phase 1.

Requirements and Restrictions

You cannot enter this command when the node control block (NCB) reorganization function is running.

Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►─ZRECP──┬───────────────────┬─ ─LEVEL-─┬─PAUSE─┬──────────────►◄
          ├─ ─PROC── ─cpuid─┤           └─xx────┘
          └─ ─ALL───────────┘
```

PROC
specifies a processor, where:

cpuid
is the ID of the processor for which you want to pause or continue recoup phase 1.

ALL
specifies that you want to pause or continue recoup phase 1 on all processors.

LEVEL
specifies one of the following:

PAUSE
places recoup phase 1 in a paused or deferred state.

xx is the maximum number of active recoup entry control blocks (ECBs) allowed during recoup phase 1. Use this value to restart recoup phase 1 after pausing it.

Additional Information

- If you restart recoup while it is in a paused state, the entry level remains 0. Therefore, you must reset the ECB level by entering **ZRECP LEVEL-***xx* before recoup can resume processing.
- Depending on the system configuration, creating many ECBs can have a negative impact on TPF system performance.
- If recoup phase 1 is started on a secondary processor, it must complete chain chasing the record structure it is processing. Pausing a secondary processor does not cause recoup to switch to another processor.
- Messages that are sent to a secondary processor are also sent to the primary processor with a processor ID to identify from where the message came.

Examples

The following example pauses recoup processing.

```
User:   ZRECP LEVEL-PAUSE

System: RECP0000I 15.23.27 RECP LEVEL COMPLETE
        RECP0000I 15.23.27 T-010  C-100  MAX ENT-00 DIFF-01 SE-00 I-5
          YD,'E8C4'   PROCESSING
        00002400 ENTRY TOTAL
        00002399 COMPLETED
        00000000 FIXED ERRORS
        00000020 FLT OR ORDINAL
        00000003 LAST SEQ # ON RCP
        BSS1     SSU IN-PROG
             C   ACTIVE PROCESSOR
             B   PRIMARY PROCESSOR
```

The following example allows a maximum of three entry control blocks (ECBs) to be processed concurrently in recoup phase 1.

```
User:   ZRECP LEVEL-3

System: RECP0000I 15.23.27 RECP LEVEL COMPLETE
        RECP0000I 15.23.27 T-010  C-100  MAX ENT-03 DIFF-01 SE-00 I-5
          YD,'E8C4'   PROCESSING
        00002400 ENTRY TOTAL
        00002399 COMPLETED
        00000000 FIXED ERRORS
        00000020 FLT OR ORDINAL
        00000003 LAST SEQ # ON RCP
        BSS1     SSU IN-PROG
             C   ACTIVE PROCESSOR
             B   PRIMARY PROCESSOR
```

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP NOREBUILD–Bypass Rebuild of Rollin Directories

Use this command to bypass the rebuilding of the rollin directories used for the second recoup rollin.

Requirements and Restrictions

- You must use this command if you are running recoup phase 3 in 1052 state or if you are using the ADR tape.
- You must use this command if error message BCP00024E was issued during the previous step.
- This command is valid only after ZRECP PROTECT or ZRECP IGNORE command processing is completed successfully.

Format

```
►►──ZRECP NOREBUILD──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example bypasses the rebuilding of the rollin directories.

Note: The three examples shown are actually part of the same display.

```
User:   ZRECP NOREBUILD

System: BCP00035I 12.45.10 RECOUP LOST ADDRESS REBUILD COMPLETE
        RECP0100I 12.45.10
        POOL     ACTIVITY DURING RECOUP    PSEUDO DIRECTORY COUNTS
        ---- ------------------------- -------------------------
         SLT                         0                         0 _
         SDP                         0                   223 302
         LLT                         0                         0
         LDP                         0                   371 167
         4LT                         0                         0
         4DP                         0                   431 956 _
         4D6                         0                    67 778
        END OF DISPLAY+
```

```
CSMP0097I 12.45.10 CPU-B SS-BSS  SSU-HPN  IS-01
BRTV0003I 12.45.10 RECOUP ACTIVITY COUNTS
POOL COUNT
TYPE TYPE                         COUNTS
-------------------------------------
 SDP  TOT                        225 500
      ---
      GFS                              0
      INU                          2 198
      IDC                              0
      EXL                              0
      EAU                              0
      PSU                        223 302
      ---
      DEA                              0
      AVL                        221 650
      ROL                          1 652
      ---
      SKP                              0
      REL                              0
      LOS                          1 652
-------------------------------------
 LDP  TOT                        375 540
      ---
      GFS                              0
      INU                          4 373
      IDC                              0
      EXL                              0
      EAU                              0
      PSU                        371 167
      ---
      DEA                              0
      AVL                        367 851
      ROL                          3 316
      ---
      SKP                              0
      REL                              0
      LOS                          3 316
-------------------------------------
 4DP  TOT                        432 392
      ---
      GFS                              0
      INU                            436
      IDC                              0
      EXL                              0
      EAU                              0
      PSU                        431 956
      ---
      DEA                              0
      AVL                        428 814
      ROL                          3 142
      ---
      SKP                              0
      REL                              0
      LOS                          3 142
-------------------------------------
 4D6  TOT                         67 800
      ---
      GFS                              0
      INU                             22
      IDC                              0
      EXL                              0
      EAU                              0
      PSU                         67 778
      ---
      DEA                              0
      AVL                         67 776
      ROL                              2
      ---
      SKP                              0
      REL                              0_
      LOS                              2
END OF DISPLAY+
```

```
CSMP0097I 12.45.11 CPU-B SS-BSS  SSU-HPN  IS-01
BCP00019A 12.45.11 DESTRUCTIVE SEGMENT RESPOND

      ENTER ZRECP PROCEED TO ROLLIN

                       OR
   ZRECP SKIP/ZRECP ABORT TO BYPASS ROLLIN  _
CSMP0097I 12.45.11 CPU-B SS-BSS  SSU-HPN  IS-01
DFPC0011I 12.45.10 24MAY CYD0  AVAILABLE FILE POOL COUNTS
                               FILE      CORE      ORD
SST DEVA                    323 070       756 00000000
SDP DEVA                    220 550      1100 000000D1 _
LST DEVA                     91 476       462 00000131
LDP DEVA                    366 570      1281 0000017E
4ST DEVA                     92 016       240 00000249
    DEVB                     73 200        24 000002F4
    TOT                     165 216       264
4DP DEVA                    224 272     15739 000002A1 _
    DEVB                    182 914      5793 00000346
    TOT                     407 186     21532
4D6 DEVA                     24 240       480 0000039A
    DEVB                     42 600       456 000003A9
    TOT                      66 840       936
END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP OFLMR–Display Recoup Offline Multiple Releases

Use this command to display all offline multiple releases from applications during recoup phases 2 and 3. The addresses displayed are both pre-recoup releases and during-recoup releases.

Requirements and Restrictions

You can enter this command only after the ZRECP RESUME command has been entered.

Format

```
           ┌─580─────┐
►►─ZRECP OFLMR─┼─────────┼──────────────────────────────►◄
           └─displim─┘
```

displim
    is the number of addresses, from 1 to 580, with offline multiple releases that can be displayed.

Additional Information

None.

Examples

The following example displays all offline multiple releases from applications during recoup phases 2 and 3.

```
User:   ZRECP OFLMR

System: BOFK0008I 12.18.30
        RECOUP - OFFLINE MULTIPLE RELEASE ANALYSIS
            FILE ADR     RID  PROG
        00000000000E0EE1 JP
        00000000000E0EE5 JP
        00000000000E0EE9 JP
        00000000000E0EED JP
               :
               :
        0000000000090730 JR        _
        0000000000090730 JR
        RELEASES DISPLAYED =  508
        FACZC ERRORS FOUND =    0+
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP ONEL–Online Error Log

Use this command to display the errors found during the chain chase portion of recoup phase 1.

Requirements and Restrictions

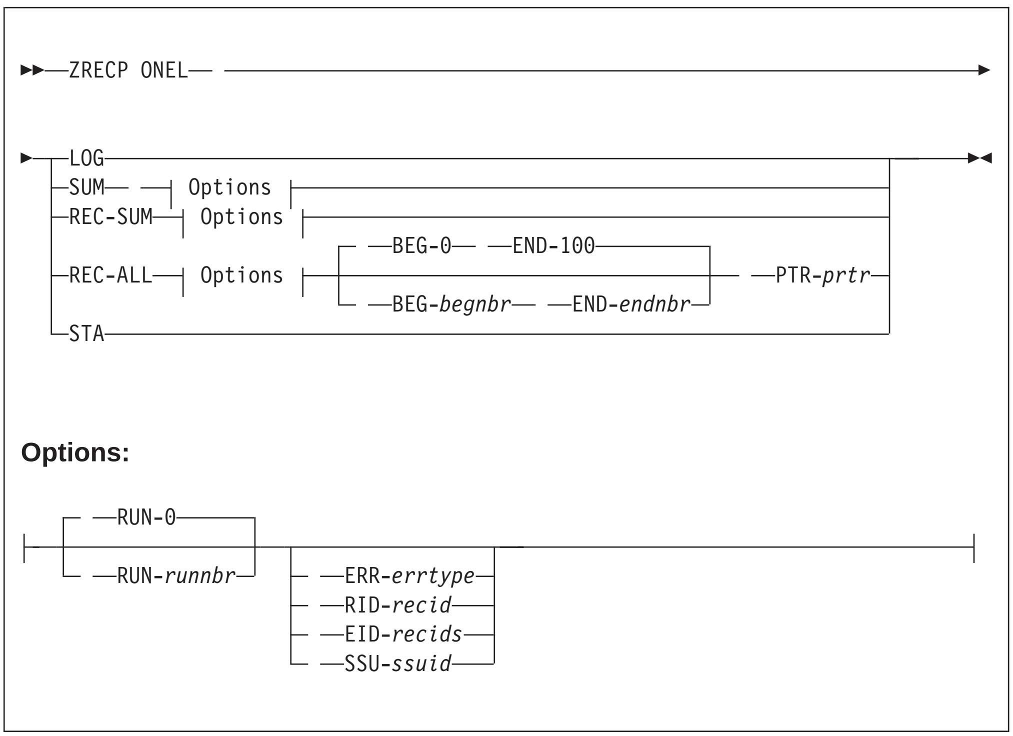
- You can enter this command only if you have the TPFDF product installed.
- You cannot use this command unless you have entered ZRECP ELOG ON to turn on error logging.
- The following settings are in segment BEWF:

  MAX_CLINE 200
  The maximum number of lines that are displayed on a visual display device.

  MAX_PLINE 1500
  The maximum number of lines that are displayed on a printer.

  MAX_DEFER 200
  The maximum number of lines that are displayed before a DEFRC macro is issued.

Format



LOG
displays a log of the error reports by recoup run number.

SUM
displays a summary of total error counts by error type.

REC-SUM
displays a summary of total error counts by record ID.

REC-ALL
displays the details of all the individual errors.

BEG-*begnbr*
specifies the beginning line number for the display, where *begnbr* is a decimal number.

END-*endnbr*
specifies the ending line number for the display, where *endnbr* is a decimal number.

PTR-*prtr*
routes the display to a printer.

STA
displays the status of the online error log (On or Off). See "ZRECP ELOG–Turn Error Log On or Off" on page 1163 for more information about turning on or turning off the online error log.

RUN-*runnbr*
displays the specified recoup run number, where *runnbr* is a number from 0 to 9. If the recoup run number is 0, the current recoup run is displayed.

ERR-*errtype*
displays the specified error type, where *errtype* is one of the following:

FXD
Fixed errors.

HDW
Hardware errors.

RCC
Record code check errors.

RID
Record ID errors.

SFW
Software errors.

TIM
Timeout errors.

RID-*recid*
displays the data for the specified record ID, where *recid* is the record ID.

Note: If omitted, all record IDs are displayed.

EID-*recids*
excludes the specified record IDs from being displayed, where *recids* specifies one or more record IDs separated by a slash (/).

Note: A maximum of four record IDs can be excluded.

SSU-*ssuid*
displays data for the specified subsystem user (SSU), where *ssuid* is the ID of the SSU.

Note: If omitted, data for all SSUs is displayed.

Additional Information

The online error log and the ID counts history records are kept synchronized by the recoup run number.

Examples

The following example displays a log of the error reports by recoup run number.

```
User:   ZRECP ONEL LOG

System: RECP0404I 11.05.43 ONLINE ERROR LOG DISPLAY  FOR  LOG
        RUN NUMBER            RUN ID         RUN DATE
               0                  10           05Aug
               1                  11           05Aug
               2                  12           05Aug
               3                  13           05Aug
               4                  14           05Aug
               5                  15           05Aug
               6                  16           07Aug     (CURRENT)
               7                   7           05Aug
               8                   8           05Aug
               9                   9           05Aug
        END OF DISPLAY
```

The following example displays a summary of total record ID errors.

```
User:   ZRECP ONEL REC-SUM

System: RECP0395I 11.07.23 ONLINE ERROR LOG DISPLAY  FOR  07Aug
        TOTAL ERRORS FOR ID JB(D1C2) -      1
        TOTAL ERRORS FOR ID JF(D1C6) -      1
        TOTAL ERRORS FOR ID JO(D1D6) -      1
        END OF DISPLAY
```

The following example displays the details of all the individual errors for a record ID, where:

REFERENCE FROM ID
: is the record ID of the record that found an error with the next address in the recoup chain.

REFERENCE FROM RCC
: is the record control check (RCC) of the record that found an error with the next address in the recoup chain.

  Note: The record version is displayed for timeout (TIM) and fixed errors (FXD).

REFERENCE FROM FILE ADR
: is the file address of the record that found an error with the next address in the recoup chain.

REFERENCE TO ID
: is the record ID of the record that was sought by the *referenced from* record.

REFERENCE TO RCC
: is the record control check (RCC) of the record that was sought by the *referenced from* record.

REFERENCE TO FILE ADR
: is the file address of the record that was sought by the *referenced from* record.

ERR ID/RC
: (for ID or RCC errors only) is the record ID or record control check (RCC) of the record that was found by the *referenced from* record.

ERR TYPE
: is the type of error (RID, RCC, HDW, SFW, or TIM).

ERR VSN
: is the version number (for FXD or TIM errors only) or the error indicator (for other type errors).

```
User:   ZRECP ONEL REC-ALL RID-J2

System: RECP0397I 12.03.15 ONLINE ERROR LOG DISPLAY  FOR  15Jun
        REFERENCE FROM           REFERENCE TO                ERR   ERR
        ID  RCC   FILE ADDRESS   ID  RCC   FILE ADDRESS  ID/RC TYP VSN
        D1F2 00 00000000000C33A1 D1F2 00 00000000000C33A5 C3C1 RID 002
        TOTAL ERRORS FOR ID J2(D1F2) -      1
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP PROCEED–Update Online Directories

Use this command to update the online pool directories with all the pool files that previous recoup phases determined to be lost or released by applications. This final step of the recoup run makes more pool files available to the TPF system.

Notes:

1. Make sure that all pool files being made available to the TPF system are no longer being used. If any pool file being made available is still in use, database corruption will occur.
2. If a bad rollin was performed and pool directory update (PDU) has not been run since the rollin, you can restore online pool directories to the state they were in before the rollin (adjusted for in-use directories) by entering **ZRDIR START RESTORE**. Do this only within 24 hours of the rollin; otherwise, severe pool depletion can result.
3. If a descriptor was missing from the recoup run, selective recoup can be run to protect its associated pool files. See the ZRECP START, ZRECP RECALL, and ZRECP PROTECT commands for information about the SEL parameter and selectively protecting a database.

Requirements and Restrictions

You can enter this command only after ZRECP REBUILD or ZRECP NOREBUILD command processing has completed successfully.

Format

```
►►──ZRECP PROCEED──────────────────────────────────────────────────►◄
```

Additional Information

- If a TPF system IPL occurs during this step, it is still possible to enter the ZRECP RESTART command to complete the rollin.
- If an error occurs during this step, you can enter the ZRECP ABORT command to update the pool counts with the new counts from the directories that were rolled in successfully.
- The following recoup activity information is displayed before the rollin operation.

  Note: The three examples shown are actually part of the same display.

```
System: RECP0100I 12.45.10
        POOL     ACTIVITY DURING RECOUP    PSEUDO DIRECTORY COUNTS
        ---- -------------------------- --------------------------
         SLT                          0                          0 _
         SDP                          0                    223 302
         LLT                          0                          0
         LDP                          0                    371 167
         4LT                          0                          0
         4DP                          0                    431 956 _
         4D6                          0                     67 778
        END OF DISPLAY+
```

```
System: BRTV0003I 12.45.10 RECOUP ACTIVITY COUNTS
        POOL COUNT
        TYPE TYPE                          COUNTS
        -------------------------------------
         SDP  TOT                         225 500
              ---
              GFS                               0
              INU                           2 198
              IDC                               0
              EXL                               0
              EAU                               0
              PSU                         223 302
              ---
              DEA                               0
              AVL                         221 650
              ROL                           1 652
              ---
              SKP                               0
              REL                               0
              LOS                           1 652
        -------------------------------------
         LDP  TOT                         375 540
              ---
              GFS                               0
              INU                           4 373
              IDC                               0
              EXL                               0
              EAU                               0
              PSU                         371 167
              ---
              DEA                               0
              AVL                         367 851
              ROL                           3 316
              ---
              SKP                               0
              REL                               0
              LOS                           3 316
        -------------------------------------
         4DP  TOT                         432 392
              ---
              GFS                               0
              INU                             436
              IDC                               0
              EXL                               0
              EAU                               0
              PSU                         431 956
              ---
              DEA                               0
              AVL                         428 814
              ROL                           3 142
              ---
              SKP                               0
              REL                               0
              LOS                           3 142
        -------------------------------------
         4D6  TOT                          67 800
              ---
              GFS                               0
              INU                              22
              IDC                               0
              EXL                               0
              EAU                               0
              PSU                          67 778
              ---
              DEA                               0
              AVL                          67 776
              ROL                               2
              ---
              SKP                               0
              REL                               0_
              LOS                               2
        END OF DISPLAY+
```

```
System: CSMP0097I 12.45.11 CPU-B SS-BSS  SSU-HPN  IS-01
        BCP00019A 12.45.11 DESTRUCTIVE SEGMENT RESPOND

              ENTER ZRECP PROCEED TO ROLLIN

                               OR
           ZRECP SKIP/ZRECP ABORT TO BYPASS ROLLIN  _
        CSMP0097I 12.45.11 CPU-B SS-BSS  SSU-HPN  IS-01
        DFPC0011I 12.45.10 24MAY CYD0  AVAILABLE FILE POOL COUNTS
                                      FILE      CORE      ORD
        SST DEVA                   323 070       756 00000000
        SDP DEVA                   220 550      1100 000000D1 _
        LST DEVA                    91 476       462 00000131
        LDP DEVA                   366 570      1281 0000017E
        4ST DEVA                    92 016       240 00000249
            DEVB                    73 200        24 000002F4
            TOT                    165 216       264
        4DP DEVA                   224 272     15739 000002A1 _
            DEVB                   182 914      5793 00000346
            TOT                    407 186     21532
        4D6 DEVA                    24 240       480 0000039A
            DEVB                    42 600       456 000003A9
            TOT                     66 840       936
        END OF DISPLAY
```

The information in the previous example allows you to determine whether to abort, skip, or proceed with the recoup rollin, where:

TOT
: is the total number of allocated pool files.

GFS
: is the number of pool files dispensed during recoup.

INU
: is the number of pool files found by recoup phase 1. These are from the VFA-resident pseudo directories.

IDC
: is the number of pool files found by recoup phase 1. These are from the ID counts database. This information will only be displayed if recoup phase 1 was run while the TPF system was above 1052 state, and only if TPFDF support is present.

EXL
: is the number of pool files excluded by ZRECP REBUILD command processing.

EAV
: is the number of phase 3 erroneously available pool files.

PSU
: is the number of pool files that are not being used.

DEA
: is the total number of deactivated pool files.

AVL
: is the total number of available pool files at the end of chain chase processing.

ROL
: is the number of new pool files to make available (for example, rollin). This is approximately how many new pool files will be returned after a ZRECP PROCEED command is run.

Notes:

1. ROL equals PSU−DEA−AVL.
2. ROL equals LOS+SKP+REL.

SKP
: is the number of new pool files that have been skipped during system cycle-up.

REL
: is the number of new pool files that have been released by applications.

LOS
: is the number of lost pool files that have been recovered.

Examples

The following example updates online pool directories with all the pool files that previous recoup phases have determined to be lost or released by applications.

```
User:   ZRECP PROCEED

System: BCP00047I 12.48.45 RECOUP REMOVAL OF CAPTURE2 FROM ROLLIN STARTED_
        BCP00048I 12.48.48 RECOUP REMOVAL OF CAPTURE2 FROM ROLLIN COMPLETE_
        BCP00028I 12.48.50 RECOUP SONRI ROLLIN STARTED_
        BCP20008I 12.48.58 DIRECTORY UPDATE/KEYPOINT 9 UPDATE COMPLETE+
        CSMP0097I 12.48.58 CPU-B SS-BSS  SSU-HPN  IS-01
        BCPE0010I 12.48.58 BCPU - END OF DIRECTORY UPDATE

        ADDRESSES RETURNED

        POOL  DEV
        TYPE  TYPE                      COUNTS
        ----  ----  --------------------------
        SDP   DEVA                       1 650
        LDP   DEVA                       3 314 _
        4DP   DEVA                         840
              DEVB                          62
        4D6   DEVA                           0
              DEVB                           0
        BCPE0011I 12.48.58 BCPE - END OF DIRECTORY UPDATE+
        CSMP0097I 12.48.59 CPU-B SS-BSS  SSU-HPN  IS-01
        BCP00029I 12.48.59 RECOUP SONRI ROLLIN COMPLETE_
        DFPC0011I 12.49.01 24MAY CYD0  AVAILABLE FILE POOL COUNTS
                                     FILE      CORE      ORD
        SST DEVA                  323 070       767 00000000
        SDP DEVA                  222 200      1100 000000D1 _
        LST DEVA                   91 476       462 00000131
        LDP DEVA                  369 884      1267 0000017E
        4ST DEVA                   92 016       240 00000249
            DEVB                   73 200        24 000002F4
            TOT                   165 216       264
        4DP DEVA                  225 112     15739 000002A1 _
            DEVB                  182 976      5793 00000346
            TOT                   408 088     21532
        4D6 DEVA                   24 240       480 0000039A
            DEVB                   42 600       456 000003A9
            TOT                    66 840       936
        END OF DISPLAY+
```

Related Information

See *TPF Database Reference* for more information about recoup and PDU functions and procedures.

ZRECP PROFILE–Option Table Input Message Driver

Use this command to display or set your recoup run-time options.

Requirements and Restrictions

Most options cannot be changed while recoup is running.

Format

```
>>──ZRECP PROFILE──┬─DISPLAY──────────────┬──────────────────────────><
                   └─┤ Recoup Options ├───┘

Recoup Options:

   ┌─RCP-NONE───────────────────────────┐ ┌─ERRLOG-RCP──┐
├──┼────────────────────────────────────┼─┼─────────────┼──────────────>
   ├─RCP-ALL────────────────────────────┤ └─ERRLOG-BOTH─┘
   └─┬─RCP-charids─┬─┬─RCPX-hexids─┬────┘
     └─────────────┘ └─────────────┘

   ┌─DIRTIM-500────┐ ┌─ADR-NO──┐ ┌─ADRNUM-32─────┐
>──┼───────────────┼─┼─────────┼─┼───────────────┼─────────────────────>
   └─DIRTIM-dirtim─┘ └─ADR-YES─┘ └─ADRNUM-adrnum─┘

   ┌─BCHMAX-100────┐ ┌─FIXERMAX-100──────┐
>──┼───────────────┼─┼───────────────────┼─────────────────────────────>
   └─BCHMAX-bchmax─┘ └─FIXERMAX-fixermax─┘

   ┌─EALOSTMAX-1000──────┐ ┌─CSERRMAX-100──────┐
>──┼─────────────────────┼─┼───────────────────┼───────────────────────>
   └─EALOSTMAX-ealostmax─┘ └─CSERRMAX-cserrmax─┘

   ┌─CSTIMEOUT-300───────┐ ┌─STATUSCON-YES─┐ ┌─REFFMMAX-1000───┐
>──┼─────────────────────┼─┼───────────────┼─┼─────────────────┼───────>
   └─CSTIMEOUT-cstimeout─┘ └─STATUSCON-NO──┘ └─REFFMMAX-refmax─┘

   ┌─REFFROM-NONE──────────────────┐ ┌─FSC-YES─┐
>──┼───────────────────────────────┼─┼─────────┼───────────────────────┤
   └─REFFROM-─┬─DEACTIVATION─┬─────┘ └─FSC-NO──┘
              ├─FARF3────────┤
              ├─FARF4────────┤
              └─FARF5────────┘
```

DISPLAY
displays the recoup option table.

RCP
specifies one of the following:

ALL
specifies that all IDs are to be written to the RCP tape.

NONE
specifies that no IDs are to be written to the RCP tape.

charids
is a continuous list of 1- to 10-character IDs that are to be written to the RCP tape. The following example shows the RCP parameter with two IDs in character format:

```
RCP-AABB
```

RCPX-*hexids*
specifies the specific hexadecimal IDs that are to be written to the RCP tape, where *hexids* is a continuous list of 1- to 10-hexadecimal IDs. The following example shows the RCP parameter with two IDs in hexadecimal format:

```
RCPX-0A0B0C0D
```

ERRLOG
specifies one of the following:

RCP
specifies that errors are to be written to the RCP tape.

BOTH
specifies that errors are to be written to both an online file and the RCP tape.

Note: You can specify this parameter only if you have the TPFDF product installed.

DIRTIM-*dirtim*
specifies the directory capture timeout value, where *dirtim* is the number of seconds from 00000 to 99999.

ADR
specifies one of the following:

NO
specifies that items are to be logged to a file.

Note: If the TPF system is in 1052 state during recoup processing, items are logged to the ADR tape even if NO was specified.

YES
specifies that items are to be logged to the ADR tape.

Note: You cannot change the value of this parameter at any time during recoup processing after you enter the ZRECP START command.

ADRNUM-*adrnum*
specifies the number of bytes to be written for each ADR item, where *adrnum* is a decimal value from 0000 to 9999.

Note: You cannot change the value of this parameter at any time during recoup processing after you enter the ZRECP START command.

BCHMAX-*bchmax*
specifies the number of broken chains to be logged, where *bchmax* is a decimal value from 00000000 to 99999999. This includes both fixed errors and timeout errors. Fixed errors will continue to be logged until both BCHMAX and

FIXERMAX values are exceeded. However, timeout errors are always logged regardless of whether the BCHMAX or FIXERMAX values have been exceeded.

FIXERMAX-*fixermax*
specifies the number of fixed errors to report to the computer room agent set (CRAS) per record ID, where *fixermax* is a decimal value from 0000 to 9999. This paramer also defines how many fixed errors will be written to the online error log after the BCHMAX limit is exceeded.

EALOSTMAX-*ealostmax*
specifies the maximum number of pool records used to log phase 3 erroneously available (EA) and lost pool files, where *ealostmax* is a decimal value from 00000000 to 99999999.

CSERRMAX-*cserrmax*
specifies the number of TPF collection support (TPFCS) errors to report to the CRAS for each datastore, where *cserrmax* is a decimal value from 0000 to 9999. If you specify a value of zero, recoup processing will report all errors.

CSTIMEOUT-*cstimeout*
specifies half of the maximum amount of time that TPFCS data store processing will wait after the last ECB is created before continuing, where *cstimeout* is a decimal value in seconds from 0000 to 9999. A timeout condition occurs if any ECBs have not completed normally after twice the specified amount of time after which all outstanding ECBs are cleaned up. If you specify a value of zero, recoup will not check for a timeout condition.

STATUSCON
specifies one of the following:

YES
specifies that status messages will be displayed on the console while recoup phase 1 is running.

NO
specifies that status messages will not be displayed on the console while recoup phase 1 is running.

Note: If STATUSCON-NO is specified, status messages will still be routed to the real-time database services (RDBS) function support console (FSC) unless FSC-NO is also specified.

REFFMMAX-*refmax*
specifies the number of file addresses to be logged, where *refmax* is any decimal value from 00000000 to 99999999.

REFFROM
specifies one of the following:

NONE
performs no logging.

DEACTIVATION
logs deactivation reference-from file addresses up to the maximum allowed into the SRM61A8 database.

Note: You can specify this parameter only if you have the TPFDF product installed.

FARF3
logs FARF3 reference-from file addresses up to the maximum allowed into the SRM61A8 database.

Note: You can specify this parameter only if you have the TPFDF product installed.

FARF4
logs FARF4 reference-from file addresses up to the maximum allowed into the SRM61A8 database.

Note: You can specify this parameter only if you have the TPFDF product installed.

FARF5
logs FARF5 reference-from file addresses up to the maximum allowed into the SRM61A8 database.

Note: You can specify this parameter only if you have the TPFDF product installed.

FSC
specifies one of the following:

YES
specifies that recoup status messages will be sent to the real-time database services (RDBS) function support console (FSC) that was defined by the ZACRS command.

NO
specifies that recoup status messages will not be sent to the RDBS FSC, but will be sent to the default receive-only (RO) FSC.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZRECP PROFILE HELP
  ZRECP PROFILE ?
- Each time you enter the ZRECP SETUP command, all default values are reset.

Examples

The following example sets the recoup options so that all IDs are written to the RCP tape.

```
User:   ZRECP PROFILE RCP-ALL

System: RECP0504I 13.16.59 RECOUP OPTIONS
        CURRENT OPTIONS                 LAST OPTIONS

        DIRTIM    - 00500               DIRTIM    - 00500
        ERRLOG    - BOTH                ERRLOG    - BOTH
        ADR       - NO                  ADR       - NO
        ADRNUM    - 0032                ADRNUM    - 0032
        FIXERMAX  - 0100                FIXERMAX  - 0100
        BCHMAX    - 00000100            BCHMAX    - 00000100
        EALOSTMAX - 00001000            EALOSTMAX - 00001000
        REFFMMAX  - 00001000            REFFMMAX  - 00001000
        REFFROM   - FARF4               REFFROM   - FARF4
        STATUSCON - YES                 STATUSCON - YES
        FSC       - YES                 FSC       - YES

        RCP       - ALL                 RCP       - ALL
        CURR RCPX - D1F8 D1F9
        LAST RCPX - D1F8 D1F9

        END
```

The following example sets recoup options that determine the directory capture timeout value, the maximum number of broken chains to be logged, and which records are written to the RCP tape.

```
User:   ZRECP PROFILE BCHMAX-00099999 DIRTIM-200 RCP-AA RCPX-1234

System: RECP0504I 13.16.59 RECOUP OPTIONS
        CURRENT OPTIONS                  LAST OPTIONS

        DIRTIM    - 00200                DIRTIM    - 00200
        ERRLOG    - BOTH                 ERRLOG    - BOTH
        ADR       - NO                   ADR       - NO
        ADRNUM    - 0032                 ADRNUM    - 0032
        FIXERMAX  - 0100                 FIXERMAX  - 0100
        BCHMAX    - 00099999             BCHMAX    - 00099999
        EALOSTMAX - 00001000             EALOSTMAX - 00001000
        REFFMMAX  - 00001000             REFFMMAX  - 00001000
        REFFROM   - FARF4                REFFROM   - FARF4
        STATUSCON - YES                  STATUSCON - YES
        FSC       - YES                  FSC       - YES

        RCP       - AA                   RCP       - ALL
        CURR RCPX - 1234
        LAST RCPX - D1F8 D1F9

        END
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP PROTECT–Protect the Database from Erroneous Addresses

Use this command to protect pool addresses that were marked as erroneously available during recoup phase 1 from being used by the TPF system by removing them from online pool directories. The ZRECP PROTECT command performs the first recoup phase 3 rollin.

The total number of erroneously available addresses is shown when ZRECP RESUME command processing is completed. If that number is 0, enter **ZRECP IGNORE** rather than ZRECP PROTECT. If the number is small, enter **ZRECP PROTECT**. If the number is very large, see your system programmer before responding with a command.

Requirements and Restrictions

- You cannot enter this command if a selective recoup was started or if recoup was started with the TEST1 option.
- You can enter this command only after ZRECP RESUME command processing is completed successfully.

Format

```
►►─ZRECP PROTECT─┬──────────┬──────────────────────────────────────►◄
                 ├─ ─SEL────┤
                 └─ ─EXCDUR─┘
```

SEL
: protects a database that was selectively chain chased during recoup phase 1 by the ZRECP RECALL command. This parameter can only be specified if the ZRECP RECALL command was previously entered with the SEL parameter specified.

EXCDUR
: excludes pool addresses that were released during recoup from the final recoup rollin that could make these addresses available to the TPF system. Specify this parameter only if a problem occurred with the release file storage (RFS) activity during recoup phase 1 that could cause incorrect RFS results.

Additional Information

Does not protect erroneously available addresses in core directories.

Examples

The following example protects pool addresses that were marked erroneously available during recoup phase 1 from being used by the TPF system by removing them from online pool directories.

```
System: BCP00007I 13.46.38 RECOUP BUILD ERRONEOUS AVAILABLE STARTED
        BOFF0010I 13.46.38 PSEUDO DIR FLUSH STARTED
        BOFF0011I 13.46.38 PSEUDO DIR FLUSH COMPLETE
        RECP00F6I 13.46.38 PSEUDO DIRECTORY VERIFY STARTED
        RECP00F8I 13.46.38 PSEUDO DIRECTORY VERIFY COMPLETE
        BCP10016I 13.46.39 RECOUP 57 TOTAL ERRONEOUSLY AVAILABLE ADDRESSES
        BCP00008I 15.32.07 RECOUP BUILD ERRONEOUS AVAILABLE COMPLETE
        BCP00010A 15.32.07 RECOUP ENTER ZRECP PROTECT OR IGNORE TO CONTINUE

User:   ZRECP PROTECT

System: BCP00011I 15.32.28 RECOUP PROTECTION OF ERRONEOUS AVAILABLE STARTED
        BCP20008I 15.32.28 DIRECTORY UPDATE/KEYPOINT 9 UPDATE COMPLETE
        BCP00012I 15.32.28 RECOUP PROTECTION OF ERRONEOUS AVAILABLE COMPLETE
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP REBUILD–Rebuild Rollin Directories to Protect Databases

Use this command to rebuild the directories used for the second recoup rollin to protect any pool files whose IDs are listed in the recoup exclusion table.

The recoup exclusion table is updated by the ZRECP ADD command and can be displayed by entering **ZRECP DISPLAY EXCLUDE**.

Requirements and Restrictions

- You cannot enter this command if recoup phase 3 is running in 1052 state or if using the ADR tape.
- You can enter this command only after ZRECP PROTECT or ZRECP IGNORE command processing is completed successfully.

Format

```
►►──ZRECP REBUILD──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example rebuilds the directories used for the second recoup rollin to protect any pool files whose IDs are listed in the recoup exclusion table.

```
User:   ZRECP STATUS

System: BCP00045I 09.59.51 RECOUP REBUILD OF LOST ADDRESSES IN PROGRESS
                         OR
               WAITING FOR LOST ADDR REBUILD

User:   ZRECP REBUILD

System: BCP00034I 09.59.51 RECOUP LOST ADDRESS REBUILD STARTED
        BRB40007I 09.59.51 RECP 22 TOTAL ADDRESSES EXCLUDED
        BCP00035I 09.59.51 RECOUP LOST ADDRESS REBUILD COMPLETE
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP RECALL–Recall Recoup

Use this command to recall recoup.

Requirements and Restrictions

- When starting a new recoup run, you can enter this command only after ZRECP START command processing is completed successfully.
- If running recoup phase 3 in 1052 state, you must enter this command after ZRECP RESUME command processing is completed successfully.
- You cannot enter this command when the node control block (NCB) reorganization function is running.

  Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►──ZRECP RECALL──┬──────────────────────────────────────┬──────────────►◄
                  │                ┌─ ──00──┐             │
                  └─ ──SEL──┬─ ──recid──┼────────┼──┬─────┘
                            │           ├─ ──vsn─┤  │
                            │           └─ ──ALL─┘  │
                            └─ ──DS-dsname──────────┘
```

SEL
: starts recoup phase 1 selective chain chasing. This parameter can only be specified if **ZRECP START SEL** was entered previously.

recid
: is a 2- or 4-character record type ID.

vsn
: is the version of the specified record ID.

ALL
: specifies all versions of the specified record ID are chain chased.

DS-*dsname*
: starts recoup phase 1 selective chain chasing for a specified TPFCS data store, where *dsname* is the name of a data store.

Additional Information

None.

Examples

The following example recalls recoup to start chain chase processing and prompts you to correct fixed errors that were found during chain chase processing.

ZRECP RECALL

```
User:   ZRECP RECALL

System: RECP0008I 13.23.06 SWITCHING RTA
        COTS0382I 13.23.06 TWEV BSS    TAPE RTA SWITCHED FROM 420 TO 421
                           VSN IS NOW Z00041
        BKP60001I 13.23.06
        FC33 LOGGING RECORDS SWITCHED
        RECP0003I 13.23.06 PSEUDO DIRECTORY INITIALIZATION STARTED
        COTS0300A 13.23.06 TWEV BSS    REMOVE RTA FROM DEVICE 420
                           VSN Z00040 G0049 S0001 F38K     SL  BLK    NOCOMP
        RECP0038I 13.23.06 SON DIR CAPT STARTED
        RECP0026I 13.23.07 CAPTURE FINISHED
        BRV00001I 13.23.08 PSEUDO DIRECTORY INITIALIZATION COMPLETED FOR PROC - B
        BRV00001I 13.23.08 PSEUDO DIRECTORY INITIALIZATION COMPLETED FOR PROC - C
        RECP00F6I 13.23.08 PSEUDO DIRECTORY VERIFY STARTED
        RECP00F8I 13.23.08 PSEUDO DIRECTORY VERIFY COMPLETE
        BRIE0002I 13.23.09 ID TABLE INITIALIZED FOR CPU C
        BRIE0002I 13.23.11 ID TABLE INITIALIZED FOR CPU B
        RECP0602I 13.23.11 CREATING RECOUP SCHEDULER
        RECP0001I 13.23.12 BEGIN     GROUP ID - JO,'D1D6' VSN - 000
        RECP00D2I 13.23.12 PSEUDO DIRECTORY INIT STARTED
        RECP00D4I 13.23.12 PSEUDO DIRECTORY INIT COMPLETE
        RECP0001I 13.23.12 STARTING  GROUP ID - JO,'D1D6' VSN - 000
                            FOR SSU - HPN  , PROC - B  , ISTREAM - 01
        RECP0001I 13.23.12 STARTING  GROUP ID - JO,'D1D6' VSN - 000
                            FOR SSU - HPN  , PROC - C  , ISTREAM - 01
        RECP0002I 13.23.13 T-013 C-001  MAX ENT-130 DIFF-0000 SE-00 I-01
          JO,'D1D6' VSN 000   COMPLETED
        00000060 ECB TOTAL    00000000 POOL FINDS
        00000060 COMPLETED    00000000 DUPES
        00000010 ORDINAL      00000000 RCI REF
        00000000 FIXED ERRS   00000000 ERRORS
        **NONE** RCP SEQ #    00000001 TIME
        HPN      SSU IN-PROG
            B    ACTIVE CPU
        RECP0001I 13.23.13 BEGIN     GROUP ID - JS,'D1E2' VSN - 002
        RECP00F0I 13.23.13 PSEUDO DIRECTORY MERGE STARTED
        RECP00E4I 13.23.14 PSEUDO DIRECTORY MERGE COMPLETE
        RECP0001I 13.23.14 STARTING  GROUP ID - JS,'D1E2' VSN - 002
                            FOR SSU - HPN  , PROC - B  , ISTREAM - 01
        RECP0001I 13.23.14 STARTING  GROUP ID - JS,'D1E2' VSN - 002
                            FOR SSU - HPN  , PROC - B  , ISTREAM - 02
        :
        RECP0701I 13.23.34 PROCESSING STARTING FOR DS IFSXBSS
        RECP0702I 13.23.34 PROCESSING COMPLETED FOR DS IFSXBSS
        RECP0708I 13.23.34 REST AREA-5   SSU-HPN
        DATA STORE IFSXBSS  COMPLETED
               8 ECBS STARTED             8 PIDS COMPLETED
               8 ECBS COMPLETED           0 TPFCS ERRORS
               0 ACTIVE ROOT SLOTS        0 ELAPSED TIME
        :
        RECP00FAI 13.23.37 PSEUDO DIRECTORY FLUSH STARTED
        RECP00FCI 13.23.37 PSEUDO DIRECTORY FLUSH COMPLETE
        RECP00CCW 13.23.37 UNPROCESSED RECORD IDS WILL NOW BE PROCESSED
        RECP0001I 13.23.37 BEGIN     GROUP ID - JS,'D1E2' VSN - 001
        RECP0001I 13.23.37 STARTING  GROUP ID - JS,'D1E2' VSN - 001
                            FOR SSU - HPN  , PROC - B  , ISTREAM - 01
        :
        RECP0002I 13.23.56 T-013 C-012  MAX ENT-130 DIFF-0000 SE-00 I-16
          JS,'D1E2' VSN 001   COMPLETED
        00000480 ECB TOTAL    00000000 POOL FINDS
        00000480 COMPLETED    00000000 DUPES
        00000005 ORDINAL      00000000 RCI REF
        00000000 FIXED ERRS   00000000 ERRORS
        00000002 RCP SEQ #    00000018 TIME
        HPN      SSU IN-PROG
            B    ACTIVE CPU
        :
        RECP00FAI 13.23.57 PSEUDO DIRECTORY FLUSH STARTED
        RECP00FCI 13.23.57 PSEUDO DIRECTORY FLUSH COMPLETE
        RECP0301I 13.23.57 TPF RECOUP COMPLETED ON CPU- B.
        RECP0012A 13.23.57 FIXED ERROR-RESPOND
        RECP0014A 13.23.57 DEFERING TIL SEL RECOUP COMPLETE OR CONTINUE RESPOND
```

The following example starts recoup phase 1 selective chain chase processing for version 1 of the PR record type.

```
User:   ZRECP RECALL SEL PR 001

System: CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        RECP0003I 19.31.36 PSEUDO DIRECTORY INITIALIZATION STARTED
        1:06:38 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        1:06:38 BRIE0002I 19.31.36 ID TABLE INITIALIZED FOR CPU E
        1:06:38 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        1:06:38 BRIE0002I 19.31.36 ID TABLE INITIALIZED FOR CPU C
        1:06:38 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        1:06:38 BRIE0002I 19.31.36 ID TABLE INITIALIZED FOR CPU D
        1:06:45 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        1:06:45 BRIE0002I 19.31.36 ID TABLE INITIALIZED FOR CPU B
        1:06:59 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        1:06:59 RECP00F6I 19.31.36 PSEUDO DIR VERIFY STARTED
        01:07:02 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:09 BRV00001I 19.31.36 PSEUDO DIRECTORY INITIALIZATION COMPLETE
        01:07:09 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:09 BRV00001I 19.31.36 PSEUDO DIRECTORY INITIALIZATION COMPLETE
        01:07:09 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:09 BRV00001I 19.31.36 PSEUDO DIRECTORY INITIALIZATION COMPLETE
        01:07:09 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:09 BRV00001I 19.31.36 PSEUDO DIRECTORY INITIALIZATION COMPLETE
        01:07:24 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:24 RECP00F8I 19.31.36 PSEUDO DIR VERIFY COMPLETE
        01:07:38 CSMP0097I 19.31.36 CPU-B SS-USA  SSU-US   IS-01    IMG-5
        01:07:38 RECP0001I 19.31.36 STARTING GROUP ID - PR,'D7D9' VSN - 001
```

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP RERUN–Start Recoup Phase 3 Again

Use this command to start recoup phase 3 processing again from the beginning if there were errors, or if you want to run phase 3 with a different set of recoup options.

Requirements and Restrictions

You can enter this command only after recoup phase 3 completed one of its functions successfully and is prompting you for more action.

Format

```
►►──ZRECP RERUN──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example starts recoup phase 3 processing again from the beginning of phase 3.

```
User:   ZRECP RERUN

System: CSMP0097I 19.33.16 CPU-B SS-BSS  SSU-BE   IS-01    IMG-5
        RECP00F6I 19.33.16 PSEUDO DIR VERIFY STARTED
        CSMP0097I 19.33.16 CPU-B SS-BSS  SSU-BE   IS-01    IMG-5
        BOF40010I 19.33.16 RECOUP INTEGRITY CHECKING STARTED_
        CSMP0097I 19.36.23 CPU-B SS-BSS  SSU-BE   IS-01    IMG-5
        BCP00003I 19.36.23 RECP GFS ACTIVITY ADD STARTED_
        BCP00004I 19.36.45 RECP GFS ACTIVITY ADD COMPLETE_
        BCP00005I 19.36.45 RECP PRE-RECOUP RELEASE PASS STARTED_
        ...
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP RESTART–Restart Recoup Phase 1

Use this command to restart recoup processing after a TPF system IPL.

- If a software IPL occurs during recoup phase 1 chain chase processing, the ZRECP RESTART command restarts recoup processing at the point where it was before the IPL.
- If a software IPL or error occurs during recoup phase 2, the ZRECP RESTART command restarts the entire phase 2 process.
- If a system IPL or error occurs during recoup phase 3, the ZRECP RESTART command restarts from the recoup phase 3 function that was running when recoup was interrupted.

Requirements and Restrictions

- You cannot enter this command when the node control block (NCB) reorganization function is running.

  Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►──ZRECP──┬──────────────────────┬── ──RESTART──────────────────────────────►◄
           └─ ──PROC── ──cpuid──┘
```

PROC
specifies a processor, where:

cpuid
is the ID of the processor for which you restart recoup processing.

Additional Information

- If recoup phase 1 is running on a secondary processor, it must complete chain chasing the record structure it is processing. Pausing a secondary processor does not cause recoup to switch to another processor.
- Messages that are sent to a secondary processor are also sent to the primary processor with a processor ID to identify where the message came from.
- If a hardware IPL occurs during recoup phase 1, force recoup processing to end and then start recoup again.
- Hardware IPLs cause recoup ID counts to be displayed somewhat lower than the actual pseudo directory counts.

Examples

The following example restarts recoup on the primary processor.

```
User:   ZRECP RESTART

System: RECP0001I 13.23.48 RESTART
```

The following example restarts recoup on secondary processor C.

```
User:    ZRECP PROC C RESTART

System:  RECP00C4I 18.01.09 "ZRECP PROC C RESTART" COMPLETE
         RECP0000I 18.01.09 RESTART BEING INITIATED (PROC - C)
         RECP00F6I 18.01.09 PSEUDO DIR VERIFY STARTED  (PROC - C)"
         RECP00F8I 18.01.53 PSEUDO DIR VERIFY COMPLETE  (PROC - C)"
         RECP0000I 18.01.53 RECOUP RESTARTED (PROC - C)
         RECP005EI 18.01.53 RESTARTING IN PRIME SSU  (PROC - C)"
         RECP005EI 18.01.53 RESTARTING IN PRIME SSU  (PROC - C)"
         RECP0000I 18.01.53 STARTING GROUP ID - JI,'D1C9' VSN - 000 PROC - C
```

Related Information

- See *TPF Database Reference* for more information about online file recoup.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP RESUME–Resume Recoup after Phase 2 Ends

Use this command to start recoup phase 3 processing after recoup phase 1 and 2 are completed.

Requirements and Restrictions

- You cannot enter this command if the ZRECP START command was entered with the SEL or TEST1 parameter.
- You can enter this command only after recoup phase 1 and 2 processing were completed successfully.
- You can enter this command only once during a recoup run. If you want to start recoup phase 3 processing a second time, enter **ZRECP RERUN**.

Format

```
►►──ZRECP RESUME──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example starts recoup phase 3 processing after recoup phase 1 and 2 have completed successfully.

```
User:   ZRECP RESUME

System: RECP0000I 10.02.54 PHASE1 TIME-00HRS. 01MINS.
        BCPY0001I 10.02.54  - KEYPOINT 9 AND SONRI SAVE STARTED
        BCPY0002I 10.02.55  - KEYPOINT 9 AND SONRI SAVE COMPLETE
        BCP00003I 10.02.55 RECOUP GFS ACTIVITY ADD STARTED
        BCP00004I 10.02.55 RECOUP GFS ACTIVITY ADD COMPLETE
        BCP00005I 10.02.55 RECOUP 'PRE' PHASE 1 RELEASE PASS STARTED
        BOFA0001I 10.02.56
               ONLINE PDU

        DOUBLE RELEASES   -        0
        FC33 RECS PROCESSED -      0
        CA RECS PROCESSED -        0
        STATUS  - I  (P=ACTIVE I=INACTIVE)
        BOFF0010I 10.02.56 PSEUDO DIR FLUSH STARTED
        BOFF0011I 10.02.56 PSEUDO DIR FLUSH COMPLETE
        BCP00006I 10.02.56 RECOUP 'PRE' PHASE 1 RELEASE PASS COMPLETE
        BCP00007I 10.02.56 RECOUP BUILD ERRONEOUS AVAILABLE STARTED
        BOFF0010I 10.02.57 PSEUDO DIR FLUSH STARTED
        BOFF0011I 10.02.57 PSEUDO DIR FLUSH COMPLETE
        RECP00F6I 10.02.57 PSEUDO DIRECTORY VERIFY STARTED
        RECP00F8I 10.02.57 PSEUDO DIRECTORY VERIFY COMPLETE
        BCP10016I 10.02.57 RECOUP 0 TOTAL ERRONEOUSLY AVAILABLE ADDRESSES
        BCP00008I 10.02.57 RECOUP BUILD ERRONEOUS AVAILABLE COMPLETE
        BCP00010A 10.02.57 RECOUP ENTER ZRECP PROTECT OR IGNORE TO CONTINUE
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP RETRY–Retry Recoup

Use this command to cause recoup to retry if an error was found during phase 1 or phase 3 directory captures or during the directory rollin.

Requirements and Restrictions

You cannot enter this command when the NCB reorganization function is running.

Note: You can enter the ZNNCB REORG command to cancel the node control block (NCB) reorganization function.

Format

```
►►──ZRECP RETRY──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

Recoup processing is retried in the following example.

```
User:   ZRECP RETRY

System: RECP0040I 15.23.07 RETRY
```

Related Information

- See *TPF Database Reference* for more information about online file recoup.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP SEL–Process Recoup Records

Use this command to process a fixed file record or a TPF collection support (TPFCS) collection again and include it in a recoup run.

Requirements and Restrictions

- Enter this command only in response to the RECP0012A message.
- Use this command only to correct fixed errors caused by timeouts or TPFCS errors while processing collections and recoup indexes.

Format

```
                                    ┌─ ──00──┐
►►──ZRECP SEL── ──┬─recid── ──addr──┼────────┼──┬──────────────────────►◄
                  │                 ├─ ──ALL─┤  │
                  │                 └─ ──vsn─┘  │
                  └─PID── ──pid─────────────────┘
```

recid
: is the 2- or 4- character record ID of a fixed file

addr
: is the file address of a fixed file record.

Note: If you have the TPFDF product installed, you can display the fixed errors by entering **ZRECP ONEL REC-ALL ERR-TIM** or **ZRECP ONEL REC-ALL ERR-FXD** if you are using the online error log and your errors for that ID have not exceeded the threshold limit.

00 specifies that version 0 of the descriptor container record for the given ID is chain chased.

ALL
: specifies that all versions of the descriptor container record for the given ID are chain chased.

vsn
: is the version of the descriptor container record for the given ID that is chain chased.

PID *pid*
: starts recoup phase 1 selective chain chasing for a specified TPF collection support (TPFCS) persistent identifier (PID), where *pid* is the 64-character hexadecimal PID. Trailing zeros can be omitted.

Additional Information

None.

Examples

The following example selectively adds version 2 of the QX record type to the current recoup run after receiving fixed errors during chain chase processing.

```
System: RECP0012A 19.31.36 FIXED ERROR-RESPOND
        RECP0014A 19.31.36 DEFERING TIL SEL RECOUP COMPLETE
        OR CONTINUE RESPOND.

User:   ZRECP SEL QX 006E54FB 02

System: RECP0036I 19.45.36 SELECTIVE RECOUP OF QX 006E54FB
          FIN
```

The following example selectively adds version 0 of the D7D5 record type to the current recoup run after receiving timeout errors during chain chase processing.

```
System: SNAP0001I 19.31.36 CPU-B SS-USA SSU-US IS-01
        PSW-071D1000 83C26378 PGM-BWRA CODE-I0004110B TERM-010000B
        RECOUP TIME-OUT:
        ID       AT 00904010 - D7D5
        VERSION  AT 006B523F - 00
        END OF SNAPC CONSOLE DISPLAY
        BWRA0400I 19.31.36 SELECTIVE CHAIN CHASE REQUIRED FOR: D7D5 VSN 000/283BA1C7

User:  ZRECP SEL D7D5 283BA1C7 000

System: RECP0036I 19.45.36 SELECTIVE RECOUP OF D7D5 283BA1C7
          FIN
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP SETUP–Initialize Internal Recoup Structures

Use this command to perform one-time initialization of the internal recoup structures before running recoup on a specific subsystem. These structures include the recoup scheduling control table (IRSCT) and the recoup active root tables (IRARTs).

Requirements and Restrictions

- The TPF system must have get file storage (GFS) pools enabled before you can use this command. Pools are not enabled until the system is cycled to CRAS state or above.
- You must enter this command on each subsystem once before running recoup on that subsystem.
- This command cannot be entered while recoup is active.

Format

```
►►──ZRECP SETUP──┬─────────┬──────────────────────────────────────────►◄
                 └──RESET──┘
```

RESET
turns off the recoup setup in progress indicator, allowing subsequent ZRECP START or ZRECP SETUP commands to be accepted.

Additional Information

- Each time you enter the ZRECP SETUP command, all recoup run-time options are reset to the default values.
- This command does the following:
  – Ensures that data store name IRCP*ssss* (where *ssss* is the name of the subsystem that the command is entered from) exists.
  – Assigns a name (IRSCT) to the IRSCT.
  – Assigns a name (IRART*n*, where *n* is a CPU ID) to each of the IRARTs.

Examples

In the following example recoup internal structures are initialized on the WP subsystem.

```
User:   WP/ZRECP SETUP

System: RECP0601I 18.08.59 RECOUP INITIALIZATION STARTED
        RECP0602I 18.09.05 CREATING RECOUP SCHEDULER
        RECP0603I 18.09.06 CREATING ACTIVE ROOT TABLE TEMPLATE
        RECP0604I 18.09.13 CREATING ACTIVE ROOT TABLE FOR PROC B
        RECP0604I 18.09.18 CREATING ACTIVE ROOT TABLE FOR PROC C
        RECP0604I 18.09.24 CREATING ACTIVE ROOT TABLE FOR PROC D
        RECP0604I 18.09.29 CREATING ACTIVE ROOT TABLE FOR PROC E
        RECP0604I 18.09.34 CREATING ACTIVE ROOT TABLE FOR PROC Z
        RECP0604I 18.09.39 CREATING ACTIVE ROOT TABLE FOR PROC 0
        RECP0605I 18.09.45 RECOUP INITIALIZATION COMPLETED
```

Related Information

- See *TPF Database Reference* for more information about online file recoup.

ZRECP SKIP–Bypass Directory Rollin

Use this command to allow recoup phase 3 to bypass the rollin of the directories.

Requirements and Restrictions

You can enter this command only in response to the BCP00019I system message.

Format

```
►►──ZRECP SKIP──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example allows recoup phase 3 to bypass the rollin of the directories.

```
System: BCP00019I 13.48.27 DESTRUCTIVE SEGMENT RESPOND

              ENTER ZRECP PROCEED TO ROLLIN

                               OR
           ZRECP SKIP/ZRECP ABORT TO BYPASS ROLLIN

User:   ZRECP SKIP

System: RECP0018I 13.50.43 PHASE III COMPLETED
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP START–Start Recoup Phase 1

Use this command to start recoup phase 1 on the primary or on a secondary recoup processor.

Requirements and Restrictions

- Before starting recoup processing, enter the ZRECP PROFILE command to set up your recoup options.
- You must mount an active and standby RTA tape.
- You must mount an active RCP tape.
- You cannot enter this command when the node control block (NCB) reorganization function is running.

  Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►──ZRECP──┬──START──┬─────────┬─┬──────────────────────────────►◄
           │         ├──TEST1──┤ │
           │         └──SEL────┘ │
           └──PROC──cpuid──START─┘
```

TEST1
starts recoup in test mode. Test mode is used primarily for testing a new recoup descriptor. Once in test mode, enter **ZRECP RECALL** to let recoup chain chase all records defined by the recoup descriptor and write all chained pool addresses to the released pool address (FC33) file and the RCP tape.

Notes:

1. Test mode captures the file pool directory.
2. Enter **ZRECP ABORT** after the recoup descriptor test ends.

SEL
starts recoup phase 1 selective chain chasing.

PROC *cpuid*
specifies a processor, where *cpuid* is the 1-character ID of a secondary processor on which you want to run recoup phase 1.

Notes:

1. You cannot enter this parameter once TPFDF recoup processing has started.
2. This parameter is valid only in a loosely coupled environment.
3. Run recoup phase 1 simultaneously on as many processors as you want.
4. Recoup processing on a secondary processor is controlled by the primary processor from which recoup was started.

Additional Information

- Recoup can be run in NORM state. There is no requirement to cycle any processor to 1052 state.

- Messages that are displayed in a secondary processor will also be displayed in the primary processor with the processor ID appended to the message to show from where the message came.

Examples

The following example starts recoup phase 1 on the primary processor.

```
User:   ZRECP START

System: RECP0504I 11.56.11 RECOUP OPTIONS
        CURRENT OPTIONS                  LAST OPTIONS

        DIRTIM    - 00500                DIRTIM    - 00500
        ERRLOG    - RCP                  ERRLOG    - RCP
        ADR       - NO                   ADR       - NO
        ADRNUM    - 0032                 ADRNUM    - 0032
        FIXERMAX  - 0100                 FIXERMAX  - 0100
        BCHMAX    - 00000100             BCHMAX    - 00000100
        EALOSTMAX - 00001000             EALOSTMAX - 00001000
        REFFMMAX  - 00001000             REFFMMAX  - 00001000
        REFFROM   - NONE                 REFFROM   - NONE
        STATUSCON - YES                  STATUSCON - YES
        RCP       - NONE                 RCP       - NONE
        RCPX      - NONE                 RCPX      - NONE

        END
        RECP0020A 11.56.12 ENSURE RCP TAPE MOUNTED FOR OUTPUT
        RECP004CA 11.56.12 ENSURE STANDBY RTA AND ALT TAPE MOUNTED FOR PHASE 1
```

The following example starts recoup phase 1 on secondary processor C.

```
User:   ZRECP PROC C START

System: RECP00A6I 11.58.18 "ZRECP PROC C START" COMPLETE
        RECP0080I 13.28.33 STARTING IN SECONDARY PROCESSOR  (PROC - C)"
        RECP0000I 11.58.19 BEGIN     GROUP ID - JS,'D1E2' VSN - 001
        RECP0000I 11.58.19 STARTING  GROUP ID - JS,'D1E2' VSN - 001
                            FOR SSU - HPN  , PROC - B  , ISTREAM - 01
        RECP0000I 11.59.23 T-000 C-000  MAX ENT-000 DIFF-0000 SE-00 I-07
          JS,'D1E2' VSN 001   PROCESSING
        00000378 ECB TOTAL    00001080 POOL FINDS
        00000378 COMPLETED    00000000 DUPES
        00000000 ORDINAL      00000000 RCI REF
        00000000 FIXED ERRS   00000000 ERRORS
        **NONE** RCP SEQ #    00000060 TIME
        HPN      SSU IN-PROG
            C    ACTIVE CPU
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP STATUS–Display Recoup Status

Use this command to display the status of recoup processing.

Requirements and Restrictions

You cannot enter this command when the node control block (NCB) reorganization function is running.

Note: You can enter the ZNNCB REORG command to cancel the NCB reorganization function.

Format

```
►►─ZRECP─┬────────────────────────┬─ ─STATUS──────────────────────────────►◄
         └─ ─PROC──┬─ ─cpuid─┬─┘
                   └─ ─ALL───┘
```

PROC
specifies a processor, where:

cpuid
is the 1-character ID of the processor for which you want to display recoup status.

Note: The PROC parameter can be entered only from the primary recoup processor.

ALL
specifies all processor IDs.

Additional Information

None.

Examples

The following example shows a display of the status information for recoup phase 1 when recoup restart is required.

```
User:   ZRECP STATUS

System: BRCP0321I 09.04.30 RECOUP RESTART REQUIRED
```

The following example shows a display of the status information for recoup phase 1 when recoup rebuild processing is required.

```
User:   ZRECP STATUS

System: BCP00036I 09.04.30
                 WAITING FOR LOST ADDR REBUILD
```

The following example shows a display of the status information for recoup phase 1 when recoup is in the process of copying directories or is waiting for you to enter the ZRECP PROCEED, ZRECP SKIP, or ZRECP ABORT command.

```
User:   ZRECP STATUS

System: BCP00049I 09.04.30 RECOUP REMOVAL OF CAPT2 FROM ROLLIN IN PROGRESS
                          OR
              WAITING FOR PROCEED/SKIP/ABORT
```

The following example shows a display of the status information for recoup phase 1 when a TPF collection support (TPFCS) database is being processed, where:

REST AREA
: shows the restart area being displayed.

SSU
: shows the subsystem user associated with the restart area.

DATA STORE
: shows the data store associated with the restart area and its current status (either IN PROGRESS or COMPLETED).

PID
: shows the root persistent identifier (PID) for a selective recoup.

ECBS STARTED
: shows the number of entry control blocks (ECBs) created to process the data store or selective PID.

PIDS COMPLETED
: shows the number of PIDs that have been completed for the data store or selective PID.

ECBS COMPLETED
: shows the number of ECBs associated with the data store that have been completed.

PID ERRORS
: shows the number of errors with PIDs associated with the data store or selective PID.

ACTIVE ROOT SLOTS
: shows the number of slots in the IBM recoup active root table (IRART) that are assigned to the data store or selective PID.

TIME ELAPSED
: shows the number of seconds since processing began on the data store.

ZRECP STATUS

```
User:   ZRECP STATUS

System: RECP0531I 15.09.00 RECOUP STATUS REPORT

        RECP0002I 15.02.53 T-001 C-001  MAX ENT-150 DIFF-0000 SE-00 I-01
            ,'FC2A' VSN 000   COMPLETED
        00002001 ECB TOTAL    00000000 POOL FINDS
        00002001 COMPLETED    00000000 DUPES
        00000000 ORDINAL      00000000 RCI REF
        00000000 FIXED ERRS   00000000 ERRORS
        **NONE** RCP SEQ #    00000010 TIME
        HPN      SSU IN-PROG
            B    ACTIVE CPU +

        RECP0708I 15.09.00 REST AREA-5  SSU-HPN
        DATA STORE TPFDB    IN PROGRESS
               2 ECBS STARTED            0 PIDS COMPLETED
               0 ECBS COMPLETED          0 TPFCS ERRORS
               1 ACTIVE ROOT SLOTS       1 ELAPSED TIME

        RECP0533I 15.09.00 RECOUP STATUS REPORT COMPLETED
```

Note: Not all of these fields will be active and displayed at the same time.

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.
- See *TPF ACF/SNA Data Communications Reference* for more information about the NCB reorganization function.

ZRECP STOP–End Recoup on a Secondary Processor

Use this command to end recoup on a secondary processor. This command notifies recoup that no more chain chasing is to be done on this particular processor. All recoup processing that was completed by this processor is preserved.

Any chain chasing that was in progress when recoup was stopped, is completed by the primary processor.

Notes:

1. For TPF files, the primary processor chain chases the remainder of the file that was not chain chased by the secondary processor.
2. For TPFDF files, the primary processor chain chases the entire file again.

Requirements and Restrictions

None.

Format

```
►►──ZRECP── ──PROC── ──cpuid── ──STOP──────────────────────────────────►◄
```

PROC *cpuid*
: specifies a processor, where *cpuid* is the 1-character ID of the processor for which you want to stop recoup.

Additional Information

- When you enter this command for the first time, a LEVEL-PAUSE is automatically performed on the target processor and the entire process waits for all chain chase activity to end and the following message is issued:

  ```
  STOP ISSUED AND WILL TAKE EFFECT WHEN ALL ECBS ARE COMPLETE.
  IF THIS TAKES TOO LONG, ISSUE STOP AGAIN
  ```

  If recoup processing does not stop in a reasonable amount of time, enter the ZRECP STOP command again to force recoup to end.
- When recoup processing that was interrupted by the ZRECP STOP command restarts on the primary processor, the pseudo directories from the stopped processor are merged with those of the current processor to ensure that the ID counts for that record ID will be as accurate as possible.
- Messages that are displayed in a secondary processor will also be displayed in the primary processor with the processor ID appended to the message to show from where the message came.

Examples

The following example ends recoup processing on secondary processor C.

```
User:    ZRECP PROC C STOP

System:  RECP00BEI 18.04.33 "ZRECP PROC C STOP" COMPLETE
         RECP00F4I 18.04.33 PSEUDO DIR COPY COMPLETE  (PROC - C)
         RECP00F6I 18.04.33 PSEUDO DIR VERIFY STARTED  (PROC - C)
         RECP00F8I 18.05.07 PSEUDO DIR VERIFY COMPLETE  (PROC - C)
         RECP00FAI 18.05.07 PSEUDO DIR FLUSH STARTED  (PROC - C)
         RECP00FCI 18.05.07 PSEUDO DIR FLUSH COMPLETE  (PROC - C)
         RECP00CEI 18.05.07 RECOUP RUN STOPPED IN THIS PROCESSOR (PROC - C)
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRECP VERIFY–Verify Pool Section Count with ID Count

Use this command to verify the recoup statistical ID counts with the pool type specified.

Pseudo directories are compared for a specified pool type to determine which pool ordinals were chain chased by multiple processors. The set of pseudo directories for each processor is compared to counts for the multiple chain chased pool references. When a multiple reference is found, the pool type and ordinal are stored in the TPFDF recoup verification database (SRM31A). A count is maintained for each pool ordinal found to be chain chased by more than one processor.

Requirements and Restrictions

- You can enter this command only if you have the TPFDF product installed.
- A maximum of 10 000 records can be stored in the TPFDF recoup verification database (SRM31A).

Format

```
►►──ZRECP VERIFY──┬─────────────────┬──────────────────────────────────►◄
                  └─ ──pooldevtype──┘
```

pool
: is one of the following pool types:

SLT
: Small, long-term

SST
: Small, short-term

SDP
: Small, long-term duplicate

LLT
: Large, long-term

LST
: Large, short-term

LDP
: Large, long-term duplicate

4LT
: 4-KB, long-term

4ST
: 4-KB, short-term

4DP
: 4-KB, long-term duplicate

4D6
: 4-KB, long-term FARF6 duplicate.

devtype
is a device type of A, B, C, or D.

Additional Information

None.

Examples

The following example verifies the recoup statistical ID counts for all pool types.

```
User:    ZRECP VERIFY

System:  BRV60013I 15.43.20 POOL TYPE COUNT VERIFICATION STARTED
         BRV60006I 15.43.24 POOL TYPE COUNT VERIFICATION COMPLETED
```

The following example verifies the recoup statistical ID counts for small duplicated pool (SDP) records on device type A.

```
User:    ZRECP VERIFY SDPA

System:  BRV60013I 15.43.49 POOL TYPE COUNT VERIFICATION STARTED
         BRV60006I 15.43.50 POOL TYPE COUNT VERIFICATION COMPLETED
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRFPC–Reconcile File Pool Counts

Use this command to reconcile the count of available file pool directory and keypoint records.

Requirements and Restrictions

- This command locks keypoint 9 (CY1KR).
- You can enter this command only if the TPF system is in 1052 or NORM state.

Format

```
►►—ZRFPC——————————————————————————————————————————————————————————►◄
```

Additional Information

If the TPF system is in NORM state, only associated long-term pool records are reconciled; if the TPF system is in 1052 state, all records are reconciled.

Examples

The following example reconciles the count of available file pool directory and keypoint records.

```
User:   ZRFPC

System: RFPC0001I 16.35.34  - STARTED RECONCILIATION OF SON POOL COUNTS
        CSMP0097I 16.35.34 CPU-B SS-BSS  SSU-HPN  IS-01
        RFPC0002I 16.35.34  - END OF JOB OF SON POOL RECONCILIATION
        CSMP0097I 16.35.34 CPU-B SS-BSS  SSU-HPN  IS-01
        DFPC0011I 16.35.34 26MAR CYD0  AVAILABLE FILE POOL COUNTS
                                       FILE      ORD
        SST DEVA                    190 630 00000000
        SDP DEVA                    133 102 00000027
        LST DEVA                     68 838 0000003C
        LDP DEVA                    115 261 0000005D
        4ST DEVA                     25 608 0000006E
            DEVB                      8 568 00000098
            TOT                      34 176
        4DP DEVA                    104 772 0000007D
            DEVB                      4 639 000000B2
            TOT                     109 411
        4D6 DEVA                        324 000000B4
            DEVB                     29 716 000000C1
            TOT                      30 040
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRHLD DELETE–Delete Lock Names

Use this command to delete locks from control unit or coupling facility (CF) lock tables.

Requirements and Restrictions

- Delete locks from active processors only in emergencies. No checks are performed when you enter this command and the lock is deleted. This can cause the CCSONS program to issue system error 00038. If the lock resides on a coupling facility (CF), the CCCFLC program can issue system error 004627 instead. If another processor is waiting for the lock, that processor is notified that the lock was deleted. If the lock is currently held by an entry control block (ECB) on this processor, the ECB is forced to exit, system error 0000DA is issued, and the lock is deleted from the record hold table (RHT).
- When a lock is on the CF you can only delete the lock from the processor that is holding (holder) the lock. Additionally, when the lock is on the CF and the processor is waiting for the lock, you can only delete the request for the lock from the processor that is waiting (waiter) for the lock.

Format

```
►►──ZRHLD DELete── ──┬─INACTive─────────────────────────────┬──────────────────►◄
                     ├─┬─LOCkname─┬── ──lockname── ──sda─┤
                     │ └─LN───────┘                         │
                     └─FA── ──fileaddr──┬────────────┬─────┘
                                        └─ ──VSYNC─┘
```

INACTive
: deletes all locks that are held by inactive processors.

Note: You can specify this parameter only for the basic subsystem (BSS).

LOCkname
: deletes a specific lock.

LN
: deletes a specific lock.

lockname
: is the 16-digit hexadecimal lock name.

sda
: is the 4-digit hexadecimal symbolic address of a device.

FA *fileaddr*
: deletes the lock for a specific record, where *fileaddr* is an 8- or 16-digit hexadecimal file address.

Note: To translate the file address correctly, use this parameter only for the subsystem and subsystem user (SSU) that owns the specified file address.

VSYNC
deletes the virtual file access (VFA) synchronization lock. If you do not specify this parameter, the RHT lock is deleted.

Additional Information

Enter the ZRHLD DISPLAY command to display information about the locks.

Examples

The following example shows that all the locks held by inactive processors are being deleted.

```
User:   ZRHLD DELETE INACT

System: RHLD0000I 08.40.54 LOCK DELETIONS COMPLETE
```

The following example shows lock 0F05038A43000080 for device 03E3 being deleted.

```
User:   ZRHLD DEL LN 0F05038A43000080 03E3

System: RHLD0021I 08.40.54 LOCK DELETION COMPLETE, CTL-38 MAY OCCUR
```

The following example shows the VFA synchronization lock for file address F403A805 being deleted.

```
User:  ZRHLD DEL FA F403A805 VSYNC

System:  RHLD0021I 08.40.54 LOCK DELETION COMPLETE, CTL-38 MAY OCCUR
```

Related Information

See *TPF Database Reference* for more information about CF record lock support and deleting locks.

ZRHLD DISPLAY–Display Lock Names

Use this command to display the lock names that exist in control unit lock tables.

Requirements and Restrictions

None.

Format

```
►►──ZRHLD DISplay── ──┬─ALL── ──sda──────┬───────────┬──┬──────────────►◄
                      │                  ├─ ──SL─────┤  │
                      │                  ├─ ──PR-id──┤  │
                      │                  └─ ──VSYNC──┘  │
                      ├─┬─LOCkname─┬─ ──lockname── ──sda─┤
                      │ └─LN───────┘                    │
                      ├─FA── ──fileaddr─────────────────┤
                      ├─DEVice── ──sda──────────────────┤
                      └─CONTinue────────────────────────┘
```

ALL
: displays information about all the locks for the control unit of a specified device.

sda
: is the 4-digit hexadecimal symbolic address of a device.

SL
: displays information about all the special locks.

PR-*id*
: displays information about all the locks for a specific processor, where *id* is the 1-character alphanumeric CPU ID of the processor.

VSYNC
: displays information about the virtual file access (VFA) synchronization lock.

LOCkname
: displays information about a specific lock.

LN
: displays information about a specific lock.

lockname
: is the 16-digit hexadecimal lock name.

FA *fileaddr*
: displays information about the lock for a specific record, where *fileaddr* is the 8- or 16-digit hexadecimal file address.

DEVice
: displays information about the locks for the specified device, which must be mounted and in use.

CONTinue
: continues to display the information about the locks.

Additional Information

- A *special lock* is a lock that has bit 0 of byte 4 (the fifth byte in the lock name) set, and is used by the TPF system for controlling special functions. A *normal* lock is a lock that has the same bit reset.
- If all the information about the locks cannot be displayed on the screen at one time, enter **ZRHLD DISPLAY CONTINUE** in 4 minutes to display the remaining locks.
- Enter the ZRHLD DELETE command to delete locks.

Examples

The following information is displayed in the examples:

LOCKNAME
: is the physical address of the locked record (for example, 02020075) and the number of the device where the lock is held (for example, 40).

  Note: The device number is a control byte in the format FFFD DDDD, where F is a flag and D is a device address bit.

HOLDERS
: is the CPU ID of the processor that is holding the record.

FILE ADDRESS
: is the file address of the locked record. If a general file is locked, GF is displayed. If a general data set is locked, GDS is displayed. If it is a special lock, SPECIAL is displayed. If the module, cylinder, head and record (MCHR) from the lock name cannot be translated into a file address, UNKN is displayed.

SUBSYS
: is the name of the subsystem that owns the record. If it is a special lock, UNKN is displayed.

WAITERS
: is the CPU ID of the processor that is waiting for the record. If there are no waiters, NONE is displayed.

SDA
: is the symbolic device address of the device where the record resides. Special locks always have the base control unit address assigned. If the SDA is not mounted, active, or valid, UNKN is displayed.

All the normal and special locks for the control unit of the specified device are displayed in the following example.

```
User:   ZRHLD DIS ALL EE5

System: RHLD0111I 16.03.03
            LOCKNAME         FILE ADDRESS    SUBSYS  SDA
        00720003 65000080  00000000F403A809   BSS    0EE5
          HOLDERS  B
          WAITERS  NONE
        LOCK DISPLAY COMPLETE
```

The following example displays information about the lock for the specified record.

```
User:   ZRHLD DISP FA F403A845 VSYNC

System: RHLD0111I 08.40.54
            LOCKNAME         FILE ADDRESS    SUBSYS  SDA
        00020988 65000080  00000000F403A805   BSS    0EE5
          HOLDERS  C
          WAITERS  B
        LOCK DISPLAY COMPLETE
```

The following example displays information about the specified lock.

```
User:   ZRHLD DISP LN 0002098865000000 EE5

System: RHLD0111I 08.40.54
            LOCKNAME         FILE ADDRESS    SUBSYS  SDA
        00020988 65000000  00000000F403A805   BSS    0EE5
          HOLDERS  B
          WAITERS  C
        LOCK DISPLAY COMPLETE
```

Related Information

None.

ZRIPL–CP Restart (IPL)

Use this command to perform a software initial program load (IPL) on the TPF system. A software IPL can be performed in place of a hardware IPL to prevent the associated hardware reset.

Requirements and Restrictions

None.

Format

```
►►──ZRIPL──┬──────┬──────────────────────────────────────────────►◄
           └──BP──┘
```

BP
performs the IPL and restart without cycling down the TPF system to 1052 state. When the IPL and restart are completed, the TPF system cycles to 1052 state.

Additional Information

If the TPF system is above 1052 state, it is cycled down to 1052 state before the IPL and restart are performed. When the IPL and restart are completed, the TPF system attempts to cycle up to the previous state.

Examples

A software IPL is performed in the following example.

```
User:   ZRIPL

System: RIPL0000I ZRIPL - OK
```

Related Information

See *TPF Main Supervisor Reference* for more information about restart and performing an IPL.

ZRLMT–Repeat Last LMT/OMT Message

Use this command to repeat the last long message transmitter (LMT) or output message transmitter (OMT) message to a terminal.

Requirements and Restrictions

None.

Format

```
►►──ZRLMT── ──┬─lniata─┬──────────────────────────────────────────────────►◄
              └─leid───┘
```

lniata
: is a line number, interchange address, and terminal address.

leid
: is a logical end-point identifier.

Additional Information

You can also enter the ZNRPT command to repeat the last LMT or OTM message for a particular node.

Examples

The last LMT or OMT message is repeated to the 520306 terminal in the following example.

```
User:   ZRLMT 520306

System: RLMT0001I 15.03.09 MESSAGE REPEATED TO 520306
```

Related Information

See *TPF Data Communications Services Reference* or *TPF ACF/SNA Data Communications Reference* for more information about the LMT program.

ZROUT START–Start an Application Active in the RCAT

Use this command to start an application. The application is marked as active in the routing control application table (RCAT).

Requirements and Restrictions

The TPF system must be in 1052 state or higher.

Format

```
►►──ZROUT START── ──name─────────────────────────────────────────────►◄
```

name
: is the 4-character alphanumeric name of an application.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZROUT HELP
ZROUT ?

Examples

The APPC application is started in the following example.

```
User:   ZROUT START APPC

System: ROUT0008I 12.29.00 APPLICATION APPC MADE ACTIVE
```

Related Information

See *TPF Data Communications Services Reference* for more information about the message routing package.

ZROUT STOP–Stop an Application in the RCAT

Use this command to stop an application. The routing control application table (RCAT) shows that the application is stopped.

Requirements and Restrictions

The TPF system must be in 1052 state or higher.

Format

```
►►──ZROUT STOP── ──name──────────────────────────────────────────────►◄
```

name
is the 4-character alphanumeric name of an application.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZROUT HELP
  ZROUT ?
- If the TPF system is in 1052 state, the application stops immediately. If the TPF system is above 1052 state, the application stops in 15 minutes.

Examples

The APPC application is stopped in the following example.

```
User:   ZROUT STOP APPC

System: ROUT0014I 12.29.58 APPLICATION APPC STOPPING

        ROUT0017I 12.44.00 APPLICATION APPC STOPPED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the message routing package.

ZRPDU ABORT–Abort PDU Create Processing

Use this command to force pool directory update (PDU) processing to end before it completes successfully.

Requirements and Restrictions

You cannot enter this command unless one of the following conditions exist:

- ZRPDU CREATE command processing is active.
- ZRPDU CREATE command processing has completed successfully and is waiting for PDU verify processing (ZDUPD S) to begin.

Format

```
►►──ZRPDU ABORT──┬──────┬──────────────────────────────────────────►◄
                 └─ ──BP─┘
```

BP
: forces PDU processing to end when normal ZRPDU ABORT command processing cannot.

Additional Information

None.

Examples

The following example ends PDU processing when normal ZRPDU ABORT command processing cannot.

```
User:   ZRPDU ABORT BP

System: BOFA0008W 09.17.12 PDU ABORT COMPLETE
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPDU CREATE–Create Exclusion Table and Start PDU

Use this command to create the pool directory update (PDU) exclusion table and to start PDU processing. PDU processing returns released addresses from the fixed online database to the TPF system.

Requirements and Restrictions

You cannot enter this command while recoup processing is active.

Format

```
>>──ZRPDU CREATE──────────────────────────────────────────────────────────>

>──┬───────────────────────────────────────────────────────────┬────────><
   │                                              ┌─────┐      │
   │                                              ▼     │      │
   ├──EXC── ──ddmmm── ──starttime── ──hhmm── ──┬──cpuid─┴─┬──┤
   │                                           └──ALL─────┘  │
   └──NOSWITCH──────────────────────────────────────────────────┘
```

EXC
excludes addresses that were released during a specified time period from being returned to the system.

ddmmm
is the starting date to exclude addresses from being returned to the system, where *dd* is the day and *mmm* is the abbreviation for the month. For example, `04NOV`.

starttime
is the starting time, by hour and minute, to exclude addresses from being returned to the system.

hhmm
is the duration, in hours and minutes, to exclude addresses from being returned to the system. You can specify a maximum limit of 2359.

cpuid
specifies a CPU ID.

ALL
specifies all CPU IDs.

NOSWITCH
prevents the fixed online database from being switched.

Note: This parameter prevents some released addresses from being returned to the TPF system.

Additional Information

None.

Examples

The following example starts PDU processing when all FC33 ordinals are being used.

```
User:   ZRPDU CREATE

System: BOFA0011I 09.54.18
        ONLINE PDU CREATE FUNCTION STARTED
        BLOG0001W 09.54.18 ALL FC33 ORDINALS IN USE - RUN ZRPDU WITH NO SWITCH
        BOFA0004E 09.54.18
        CANT SWITCH BLOG ORDINALS


User:   ZRPDU CREATE NOSWITCH

System: BOFA0011I 09.54.36
        ONLINE PDU CREATE FUNCTION STARTED_
        BOF40015I 09.54.37 PSEUDO DIR RECONCILE STARTED_
        BOF40016I 09.54.37 PSUEDO DIR RECONCILE COMPLETE_
        BOFF0010I 09.54.37 PSEUDO DIR FLUSH STARTED+
        CSMP0097I 09.54.37 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF20013I 09.54.37 PDU    - POOLS RETURNED SUMMARY DISPLAY

        POOL                         TOTALS                  RETURNED
        -------- -------------------------- --------------------------
        SST DEVA                    323 840                         0 _
        SDP DEVA                    225 500                       489
        LST DEVA                     91 938                         0
        LDP DEVA                    375 540                     2 742
        4ST DEVA                     92 256                         0
            DEVB                     73 224                         0 _
        4DP DEVA                    241 112                        54
            DEVB                    191 280                         7
        4D6 DEVA                     24 720                         0
            DEVB                     43 080                         0
        END DISPLAY
        CSMP0097I 09.54.37 CPU-A SS-BSS  SSU-BSS  IS-01
        BOFF0011I 09.54.37 PSEUDO DIR FLUSH COMPLETE+
        CSMP0097I 09.54.37 CPU-A SS-BSS  SSU-BSS  IS-01
        BOFA0018I 09.54.37
         ONLINE PDU

        DOUBLE RELEASES - 3 269
        FC33 RECS PROCESSED - 0
        CA RECS PROCESSED - 56
        STATUS - I (P=ACTIVE I=INACTIVE) _
        BOFA0006I 09.54.37
        PDU CA RELEASE ITEMS ALL PROCESSED
```

The following example starts PDU processing for released pool addresses, excluding pool addresses that were released from 03:00 AM for 4 hours, (that is from 03:00 AM to 07:00AM) on May 12 on all processors.

```
User:   ZRPDU CREATE EXC 12MAY 0300 0400 ALL

System: BOFA0011I 09.24.35
        ONLINE PDU CREATE FUNCTION STARTED
        COSL0081A 09.24.36 TWEV BSS    MOUNT STANDBY RTA TAPE
        BOFA0007I 09.24.36
        SWITCH FC33 LOGGING RECDS COMPLETE
        BOF40015I 09.24.36 PSEUDO DIR RECONCILE STARTED
        BOF40016I 09.24.36 PSUEDO DIR RECONCILE COMPLETE
        BOFF0010I 09.24.36 PSEUDO DIR FLUSH STARTED
        CSMP0097I 09.24.36 CPU-A SS-BSS  SSU-BSS  IS-01
        BOF20013I 09.24.36 PDU    - POOLS RETURNED SUMMARY DISPLAY

        POOL                       TOTALS                  RETURNED
        -------- ------------------------- -------------------------
        SST DEVA                  323 840                         0 _
        SDP DEVA                  225 500                       489
        LST DEVA                   91 938                         0
        LDP DEVA                  375 540                     2 742
        4ST DEVA                   92 256                         0
            DEVB                   73 224                         0 _
        4DP DEVA                  241 112                        54
            DEVB                  191 280                         7
        4D6 DEVA                   24 720                         0
            DEVB                   43 080                         0
        END DISPLAY
        CSMP0097I 09.24.46 CPU-A SS-BSS  SSU-BSS  IS-01
        BOFF0011I 09.24.46 PSEUDO DIR FLUSH COMPLETE+
        CSMP0097I 09.24.46 CPU-A SS-BSS  SSU-BSS  IS-01
        BOFA0018I 09.24.46
         ONLINE PDU

        DOUBLE RELEASES - 3 269
        FC33 RECS PROCESSED - 0
        CA RECS PROCESSED - 56
        STATUS - I (P=ACTIVE I=INACTIVE) _
        BOFA0006I 09.24.47
        PDU CA RELEASE ITEMS ALL PROCESSED
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPDU DISP–Display the PDU Create Exclusion Table

Use this command to display the pool directory update (PDU) exclusion table, which contains the intervals that were excluded with the ZRPDU CREATE command.

Requirements and Restrictions

None.

Format

```
►►──ZRPDU DISP──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example shows all intervals that will be excluded during PDU rollin processing.

```
User:   ZRPDU DISP

System: BOF80003I 09.26.10
        NO  DATE  START  DURATION CPU          FILEADDR
        01 23AUG   0300    0145   ALL  00000000004F031C
        02 23AUG   0745    0200    B   0000000000000000
        03 23AUG   1500    0030    B   003E00000001031C
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPDU OFLMR–Display PDU Offline Multiple Releases

Use this command to display all program segments that had offline multiple releases during pool directory update (PDU) preparation (running the ZRPDU CREATE command).

Requirements and Restrictions

You can enter this command only after ZRPDU CREATE command processing ends successfully.

Format

```
            ┌─580─────┐
►►─ZRPDU OFLMR─┼─────────┼──────────────────────────────────►◄
            └─displim─┘
```

displim
is the number of addresses, from 1 to 580, with offline multiple releases that can be displayed.

Additional Information

None.

Examples

The following example shows all program segments that had offline multiple releases during pool directory update (PDU) preparation (running the ZRPDU CREATE command).

```
User:   ZRPDU OFLMR

System: BOFK0008I 10.58.55
        RECOUP - OFFLINE MULTIPLE RELEASE ANALYSIS
             FILE ADR     RID   PROG
        000000008009B4EE  OM    CVE4
        000000008009B52E  FCC1  CYYY
        003E0000001C002E  PR    PRSR
        RELEASES DISPLAYED =    6
        FACZC ERRORS FOUND =    0
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPDU PURGE–Purge Items from the PDU Exclusion Table

Use this command to purge one item or all items from the pool directory update (PDU) exclusion table. The PDU exclusion table is created by the ZRPDU CREATE command and can be displayed by entering ZRPDU DISP.

Requirements and Restrictions

None.

Format

```
►►──ZRPDU PURGE── ──┬─ALL───┬──────────────────────────────────────────►◄
                    └─SETnn─┘
```

ALL
purges all items in the PDU exclusion table.

SET*nn*
purges only set *nn*, where *nn* is the number of the set.

Additional Information

None.

Examples

The following example shows the PDU exclusion table before and after purging interval number 3.

```
User:   ZRPDU DISP

System: BOF80003I 09.26.10
        NO   DATE  START  DURATION    CPU      FILEADDR
        01  12APR   0300    0100      ALL      00000000
        02  12MAY   0300    0400      ALL      00000000
        03  23AUG   0200    0100      ALL      00000000
        END OF DISPLAY


User:   ZRPDU PURGE SET03

System: ZRPD0006I 09.26.20 INPUT SET PURGED


User:   ZRPDU DISP ALL

System: BOF80003I 09.26.29
        NO   DATE  START  DURATION    CPU      FILEADDR
        01  12APR   0300    0100      ALL      00000000
        02  12MAY   0300    0400      ALL      00000000
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPDU STATUS–Display the Status of ZRPDU CREATE Processing

Use this command to display the status of ZRPDU CREATE command processing.

Requirements and Restrictions

You cannot enter this command unless one of the following conditions exist:

- ZRPDU CREATE command processing is active.
- ZRPDU CREATE command processing has completed and is waiting for pool directory update (PDU) verify processing (ZDUPD S) to begin.

Format

```
►►──ZRPDU STATUS──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The following example shows that ZRPDU CREATE command processing is inactive.

```
User:   ZRPDU STATUS

System: BOFA0018I 09.26.36
               ONLINE PDU

        DOUBLE RELEASES   -        0
        FC33 RECS PROCESSED -      1
        CA RECS PROCESSED -        9
        STATUS  - I  (P=ACTIVE I=INACTIVE)
```

Related Information

See *TPF Database Reference* for more information about file pool support.

ZRPGM–Retrieve and Lock Program

Use this command to:

- Lock an online sharable file resident or core resident TPF program in storage
- Unlock a program from storage that is currently locked in storage
- Display the storage address of all locked programs
- Display the storage address of all locked versions of a program.

Requirements and Restrictions

You cannot lock a program that is private or that is already locked.

Format

```
►►──ZRPGM── ──┬─progname──┬──────────┬──┬──────────────────────────┬─┬──►
              │           ├─ ──Lock───┤  └─ ──LOAdset─┬─lsname─┬───┘ │
              │           └─ ──Unlock─┘               └─BASE───┘     │
              └─ALL───────────────────────────────────────────────────┘

   ┌─ ──Is-1──┐
►──┴─ ──Is-xx─┴──────────────────────────────────────────────────────►◄
```

progname
: is a 4-character alphanumeric program name.

Lock
: locks the program in core.

Unlock
: unlocks the program from core.

LOAdset
: specifies the loadset that contains the activated program.

  lsname
  : is the 5- to 8-character alphanumeric name of a loadset.

  BASE
  : indicates the base version of the program.

ALL
: displays all of the programs that were locked in core by using the ZRPGM command or the ZAPGM command.

Is-*xx*
: specifies the I-stream that this change affects, where *xx* is a decimal number from 1 to 16.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

  ZRPGM HELP
  ZRPGM ?

- Enter **ZRPGM** *progname* to display the storage address of a specific locked program.
- If you do not specify the LOADSET parameter for a lock or unlock request, the TPF system determines which program version to lock or unlock based on the activation number of the ECB.
- Programs allocated as file resident are locked in common blocks. If the program is not currently in a common block, it is moved to one and locked there.
- If you lock a program in storage and then deactivate the loadset that contains the program, the PAT slot for that program is cleaned up; that is, the program is removed from storage even though you never entered this command with the UNLOCK parameter.

Examples

In the following example, each version of the CVAP program is locked in core.

```
User:   ZRPGM CVAP LOCK

System: RPGM0001I 13.39.46 PROGRAM CVAP LOADSET BASE LOCKED IN CORE
        ------------------ AT ADDRESS 00A7DC30
```

The following example displays the storage address of all locked versions of the CVAP program.

```
User:   ZRPGM CVAP

System: RPGM0033I 13.14.12 VERSIONS OF PROGRAM CVAP LOCKED ON I-STREAM  1

        VERSION  LOADSET   CODE ADDR
        -------  --------  ---------
           40    BASE      00A7DC30
        END OF DISPLAY
```

The following example displays the storage address of all the programs that are locked in core on the main I-stream.

Note: The programs were locked in core by using the ZRPGM or ZAPGM command.

```
User:   ZRPGM ALL

System: RPGM0017I 13.39.46 PROGRAMS LOCKED ON I-STREAM  1

        PROGRAM  VERSION  LOADSET   CODE ADDR
        -------  -------  --------  ---------
         CHDD       IK    BASE      00174300
         CVAB       MG    JGARCIA   00202180
         CVAB       JW    GRATEFUL  0017B000
         CHUG       AB    DENVER    00170000
         CHUG       JX    BASE      00172130
        END OF DISPLAY
```

Related Information

See the RELPC macro and the GETPC macro in *TPF General Macros* for more information about locking programs in storage.

ZRREC BACKUP–Backup Release Record Global

Use this command to back up recoup global @BUSED by one indicator, zeroing out the previous indicator, and allowing release record switching.

Requirements and Restrictions

Enter this command only when global @BUSED indicates that all records are in use. This will cause previously released pool files to be lost until subsequent recoup processing.

Format

```
►►──ZRREC BACKUP──────────────────────────────────────────────────────►◄
```

Additional Information

- This command causes previously released records to be lost.
- This command causes global @BUSED to be updated.

Examples

The following example backs up recoup global @BUSED by one indicator.

```
User:   ZRREC BACKUP

System: BLOG0006I 14.03.58 FC33 ORDINALS HAVE BEEN MADE AVAILABLE TO THIS
        PROCESSOR MOST POOLS PREVIOUSLY RELEASED MAY NOW BE LOST
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRREC RESET–Reset Release Record Global

Use this command to reset recoup global @BUSED to zeros, which allows for release record switching.

Requirements and Restrictions

Enter this command only for migrating to integrated online pool maintenance and recoup support.

Format

```
►►──ZRREC RESET──────────────────────────────────────────────────────────►◄
```

Additional Information

- This command causes previously released records to be lost.
- This command causes global @BUSED to be updated.

Examples

The following example resets recoup global @BUSED to zeros.

```
User:   ZRREC RESET

System: BLOG0006I 14.03.58 FC33 ORDINALS HAVE BEEN MADE AVAILABLE TO THIS
        PROCESSOR, MOST POOLS PREVIOUSLY RELEASED MAY NOW BE LOST
```

Related Information

- See *TPF Database Reference* for more information about recoup functions and procedures.
- See *TPF Migration Guide: Program Update Tapes* for more information about migrating to integrated online pool maintenance and recoup support.

ZRREC SWITCH–Switch Release Record Structures

Use this command to switch the fixed records in which the release records are stored.

Requirements and Restrictions

Mount a standby RTA tape before you enter this command..

Format

```
►►──ZRREC SWITCH──────────────────────────────────────────────────────►◄
```

Additional Information

- This command causes the RTA tape on all processors to switch.
- This command causes globals @BUSED and @BLOG to be updated.

Examples

The following example switches the fixed records in which the release records are stored.

```
User:   ZRREC SWITCH

System: BOFA0007I 13.53.28
        SWITCH FC33 LOGGING RECORDS COMPLETE
        COTS0086I 13.53.28 TWEV BSS    TAPE RTA SWITCHED FROM 342 TO 341
        COTS0080A 13.53.28 TWEV BSS    REMOVE RTA FROM DEVICE 342
                           VSN 000000 G     S0001 D38K   SL  BLK    COMP
```

Related Information

See *TPF Database Reference* for more information about recoup functions and procedures.

ZRSTT–DASD Path and Status Verification

Use this command to replace the module verification table (MVT) or keypoint 6 (CTK6). Replace the MVT to ensure that all DASD devices are found through the same channel and control unit paths used by all other loosely coupled processors in the complex. Replace keypoint 6 to ensure that the status of all DASD devices is the same for all LC processors in the complex. You also have the option of canceling the IPL when the DASD paths or status do not match.

Requirements and Restrictions

You can enter this command only when you are prompted by the TPF system.

Format

```
►►──ZRSTT── ──┬─REP─┬──────────────────────────────────────────────────►◄
              └─CAN─┘
```

REP
: replaces the DASD paths of the complex in the module verification table (MVT) or the DASD status indicators in keypoint 6 with those of the IPLing processor.

CAN
: cancels the IPL by ending the restart ECB with a CTL-41B. This retains the DASD paths of the LC complex in the MVT or the DASD status indicators in keypoint 6.

Additional Information

None.

Examples

In the following example, keypoint 6 is replaced.

```
System: RSTT0004A 08.40.01
        TO CONTINUE RESTART AND REPLACE
        KEYPOINT 6 - ENTER :
           ZRSTT REP
        TO ABORT THE IPL - ENTER :
           ZRSTT CAN

User:   ZRSTT REP

System: RSTT0006I 08.40.01
        CTK6 FOR SS BSS  FILED ACCORDING TO OPTION
        RSTT0007I 08.40.01
        MVT  FOR SS BSS  FILED WITH REVISED DASD STATUS
```

Related Information

None.

ZRTCU–Record Type Conversion Utility

Use this command to convert a fixed record ID ***to*** and ***from*** its SYSEQ character representation and its corresponding numeric value. You can also produce a list of extents for a record type.

Requirements and Restrictions

None.

Format

```
►►──ZRTCU── ──┬─Cton──── ──string─────────┬──────────────────────►◄
              └─┬─Ntoc────┬─ ──┬─decnbr──┬─┘
                └─Extents─┘    ├─Xhexnbr─┤
                               └─hexnbrX─┘
```

Cton
: converts the character representation of the SYSEQ fixed record type to the corresponding numeric values.

Ntoc
: converts the numeric value of the SYSEQ fixed record type to the corresponding character representation.

Extents
: produces a list of MMCCHHR extent limits for the fixed record type.

string
: is the 2- to 8-byte character representation of the SYSEQ fixed record that begins with a number sign (#).

Note: You can substitute the ./. characters for the # if you are using a display device that does not have the # character on the keyboard.

decnbr
: is a decimal number from 0–2 147 483 647

X*hexnbr*
: is a hexadecimal number from X'00'–X'7FFFFFFF'.

*hexnbr***X**
: is a hexadecimal number from X'00'–X'7FFFFFFF'.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZRTCU HELP
ZRTCU ?

Examples

The character representation of the #KEYPT SYSEQ fixed record type is converted to the corresponding numeric values in the following example.

```
User:   ZRTCU CTON #KEYPT

System: RTCU0001I 09.40.59 RECORD TYPE #KEYPT CONVERTS TO DECIMAL 60, HEX 0000003C
```

The hexadecimal value of the SYSEQ fixed record type is converted back to its corresponding character representation in the following example.

```
User:   ZRTCU NTOC X3C

System: RTCU0003I 16.17.11 RECORD TYPE DECIMAL 60, HEX 0000003C CONVERTS TO #KEYPT
```

A list of extent limits for the specified fixed record is displayed in the following example.

```
User:   ZRTCU EXTENTS 60

System: RTCU0005I 16.18.29 RECORD TYPE DECIMAL 60, HEX 0000003C SPANS 2 EXTENTS
        MMCCHHR 0047 002F 0000 01 TO MMCCHHR 0047 0030 000E 0A
        MMCCHHR 0048 002F 0000 01 TO MMCCHHR 0048 0030 000E 0A
```

Related Information

See *TPF Database Reference* for more information about file address formats.

ZRTDM DISPLAY–Display Pool Overrides

Use this command to display the default values for pool record IDs that are not defined in the record ID attribute table (RIAT). These values are referred to as *RIAT pool overrides*.

Requirements and Restrictions

This command displays the RIAT pool overrides only for the current image.

Format

```
►►──ZRTDM DISplay── ──OVErride──────────────────────────────────────────►◄
```

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZRTDM HELP
ZRTDM ?

Examples

The RIAT pool overrides are displayed in the following example.

```
User:   ZRTDM DIS OVER

System: RTDM0003I 17.25.26 POOL OVERRIDES ARE --   VFALT - RIAT , VFAST - RIAT
                           RCSLT - RIAT   , RCSST - RIAT
        RTDM0002I 17.25.26 DISPLAY PROCESSING COMPLETED
```

Related Information

- See “ZRTDM MODIFY–Modify Pool Overrides” on page 1242 for a description of the values that are displayed when you enter this command.
- See *TPF System Installation Support Reference* or *TPF System Generation* for more information about the RIAT.

ZRTDM DISPLAY–Display RIAT Entries

Use this command to display a specific entry or group of entries in the record ID attribute table (RIAT).

Requirements and Restrictions

This command displays the RIAT entries only for the current image.

Format

```
►►──ZRTDM DISplay── ──┬─RECid-─┬─charid──────────────────────┬─┬─────────────────►◄
                      │        ├─hexid───────────────────────┤ │
                      │        │       ┌─,────────────┐      │ │
                      │        │       ▼              │      │ │
                      │        └─/───────┬─charid─┬───┴──/──┘ │
                      │                  └─hexid──┘            │
                      └─CTLVAL─────────────────────────────────┘
```

charid
: is a 2-character alphanumeric ID.

hexid
: is a 4-digit hexadecimal ID.

CTLVAL
: displays an 8-digit hexadecimal control value of the record ID attribute table (RIAT) control value on file.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZRTDM HELP
ZRTDM ?

Examples

RIAT entry A1 is displayed in the following example.

```
User:   ZRTDM DISP RECID-A1

System: RTDM0010I 17.29.36 RECID-A1  , VFAF-NO     , VFAP-NO     , XCP-YES
                           LOG-NO , RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-CFWD, RCSP-CFWD
                           RTP0-LSTA, RTP1-LSTB, RTP2-4STB, RTP3-NIU,  RTP4-NIU,
                           RTP5-NIU,  RTP6-NIU,  RTP7-NIU,  RTP8-NIU,  RTP9-NIU

        RTDM0002I 17.29.36 DISPLAY PROCESSING COMPLETED
```

A group of RIAT entries is displayed in the following example.

```
User:   ZRTDM DISP REC-/00E4,AA,00E5,AR/

System: RTDM0010I 17.29.36 RECID-00E4, VFAF-NO     , VFAP-NO     , XCP-NO
                           LOG-NO , RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-CFWD, RCSP-CFWD
                           RTP0-LSTA, RTP1-LSTB, RTP2-4STB, RTP3-NIU, RTP4-NIU,
                           RTP5-NIU,  RTP6-NIU,  RTP7-NIU, RTP8-NIU,  RTP9-NIU
        RTDM0010I 17.29.36 RECID-AA    , VFAF-NO     , VFAP-NO     , XCP-NO
                           LOG-NO , RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-CFWD, RCSP-CFWD
                           RTP0-SSTA, RTP1-SSTB, RTP2-NIU, RTP3-NIU,  RTP4-NIU,
                           RTP5-NIU,  RTP6-NIU,  RTP7-NIU, RTP8-NIU,  RTP9-NIU
        RTDM0010I 17.29.36 RECID-00E5, VFAF-NO     , VFAP-NO     , XCP-NO
                           LOG-NO , RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-CFWD, RCSP-CFWD
                           RTP0-LSTA, RTP1-LSTB, RTP2-NIU, RTP3-NIU,  RTP4-NIU,
                           RTP5-NIU,  RTP6-NIU,  RTP7-NIU, RTP8-NIU,  RTP9-NIU
        RTDM0010I 17.29.36 RECID-AR    , VFAF-NO     , VFAP-NO     , XCP-NO
                           LOG-NO , RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-CFWD, RCSP-CFWD
                           RTP0-4DPA, RTP1-4D6B, RTP2-4LTA, RTP3-4LTB, RTP4-NIU,
                           RTP5-NIU,  RTP6-NIU,  RTP7-NIU,  RTP8-NIU,  RTP9-NIU

        RTDM0002I 17.29.26 DISPLAY PROCESSING COMPLETED
```

Control value AE4D567C is displayed in the following example.

```
User:   ZRTDM DISP CTLVAL

System: RTDM0006I 17.11.44 FILE COPY HAS CONTROL VALUE OF AE4D567C
```

Related Information

- See “ZRTDM MODIFY–Modify RIAT Entries” on page 1244 for a description of the values that are displayed when you enter this command.
- See *TPF System Installation Support Reference* or *TPF System Generation* for more information about the RIAT.

ZRTDM MODIFY–Modify Pool Overrides

Use this command to modify the default values for pool record IDs that are not defined in the record ID attribute table (RIAT). These values are referred to as *RIAT pool overrides*.

Requirements and Restrictions

This command modifies the RIAT pool overrides only for the current image.

Format

```
                                    ┌─────────────────────────────┐
                                    │    (1)                      │
                                    ▼                             │
►►──ZRTDM MODify── ──OVErride── ────┬──────────────────────────────┴──────────►◄
                                    ├─┬─VFALt─┬───┬─DELay────┬──┤
                                    │ └─VFASt─┘   ├─IMMediat─┤  │
                                    │             ├─NO───────┤  │
                                    │             └─RIAT─────┘  │
                                    └─┬─RCSLt─┬───┬─DFW──┬──────┘
                                      └─RCSSt─┘   ├─CFWS─┤
                                                  ├─CFWD─┤
                                                  ├─RET──┤
                                                  ├─NO───┤
                                                  └─RIAT─┘
```

Notes:

1 You can also separate the parameters with the , character.

VFALt
modifies the VFA long-term pool override.

VFASt
modifies the VFA short-term pool override, where:

DELay
specifies that the record is a candidate for delayed filing.

IMMediat
specifies that the record is a candidate for immediate filing.

NO
specifies that the record is not a candidate for delayed or immediate filing.

RIAT
specifies that the override value is not used; the default RIAT entry is used instead.

RCSLt
modifies the RCS long-term pool override.

RCSSt
modifies the RCS short-term pool override, where:

DFW
assigns the record DASD fast write access.

CFWS
assigns the record cache fast write access (simplex write).

CFWD
assigns the record cache fast write access (duplex write).

RET
assigns the record retentive access.

NO
specifies that the record is not a candidate for caching.

RIAT
specifies that the override value is not used; the default RIAT entry is used instead.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZRTDM HELP
  ZRTDM ?
- If the RCSLT or RCSST parameter is currently set to CFWS and you want to change it, you must first change it to CFWD. This allows the duplex records that were not updated while CFWS was in effect to be written to DASD.

Examples

The VFALT parameter is changed for the RIAT pool overrides in the following example.

```
User:   ZRTDM MOD OVER VFALT-DELAY

System: RTDM0003I 17.25.26 POOL OVERRIDES ARE --   VFALT - DELAY, VFAST - RIAT
                           RCSLT - RIAT   , RCSST - RIAT
        RTDM0004I  MODIFY PROCESSING COMPLETED FOR IMAGE IMAGEABC
```

Related Information

- See *TPF System Generation* for a more detailed description of the RIAT pool override values.
- See *TPF System Installation Support Reference* or *TPF System Generation* for more information about the RIAT.

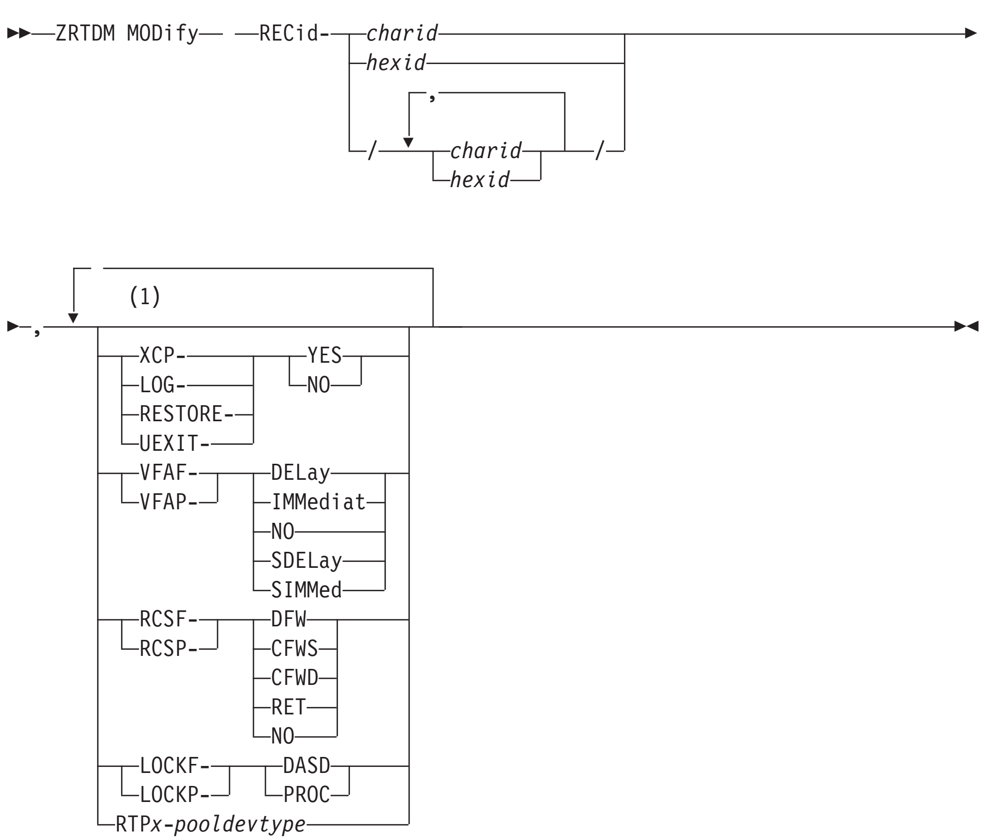
ZRTDM MODIFY–Modify RIAT Entries

Use this command to modify a specific entry or a group of entries in the record ID attribute table (RIAT).

Requirements and Restrictions

This command modifies the RIAT entries only for the current image.

Format



Notes:

1 You can also separate the parameters with the , character.

charid
is a 2-character alphanumeric ID.

hexid
is a 4-digit hexadecimal ID.

XCP
specifies if exception recording is on.

LOG
specifies if logging is on.

RESTORE
specifies if the RIAT entry is restored after a capture is performed.

Note: If you specify NO, also specify NO for the LOG and XCP parameters. Otherwise, exception recording and logging occur when the RIAT entry is not restored.

UEXIT
specifies if a user exit is defined for special processing.

VFAF
modifies the VFA fixed record attribute.

VFAP
modifies the VFA pool record attribute, where:

DELay
specifies that the record is a candidate for delayed filing.

IMMediat
specifies that the record is a candidate for immediate filing.

NO
specifies that the record is not a candidate for delayed or immediate filing.

SDELay
specifies that the record is a candidate for virtual file access (VFA) synchronization delay filing.

SIMMed
specifies that the record is a candidate for VFA synchronization immediate filing.

RCSF
modifies the record caching attribute for an RCS fixed record.

RCSP
modifies the record caching attribute for an RCS pool record, where:

DFW
assigns the record DASD fast write access.

CFWS
assigns the record cache fast write access (simplex write).

CFWD
assigns the record cache fast write access (duplex write).

RET
assigns the record retentive access.

NO
specifies that the record is not a candidate for caching.

LOCKF
modifies the locking attribute for a fixed record.

LOCKP
modifies the locking attribute for a pool record, where:

DASD
specifies that the lock is maintained by the DASD ELLF control unit.

PROC
specifies that the lock is maintained by the record hold table (RHT) of the processor.

RTP*x***-***pooldevtype*
modifies the pool characteristices for the specified record ID, where:

x is a number from 0 to 9. This number corresponds to the number specified for the RTP parameter of the GETFC macro.

pool
is one of the following pool types:

SLT
Small, long-term

SST
Small, short-term

SDP
Small, long-term duplicate

LLT
Large, long-term

LST
Large, short-term

LDP
Large, long-term duplicate

4LT
4-KB, long-term

4ST
4-KB, short-term

4DP
4-KB, long-term duplicate

4D6
4-KB, long-term FARF6 duplicate.

devtype
is a device type of A, B, C, or D.

If you do not specify a value for an RTP parameter, a value of NIU is displayed for that RTP parameter.

Attention: Changing the value specified for the RTP parameters for RIAT IDs that are in use by applications in the TPF system can cause serious problems, including system outages and database corruption.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZRTDM HELP
  ZRTDM ?
- If the RCSF or RCSP parameter is currently set to CFWS and you want to change it, you must first change it to CFWD. This allows the duplex records not updated while CFWS was in effect to be written to DASD.
- If you specify LOCKF-PROC, you cannot specify VFAF-SIMM or VFAP-SDEL.

- If you specify LOCKP-PROC, you cannot specify VFAP-SIMM or VFAP-SDEL.

Examples

RIAT entry A1 is changed in the following example.

```
User:   ZRTDM MODIFY RECID-A1,VFAF-SIMMED

System: RTDM0010I 13.47.55 RECID-A1  , VFAF-*SIMMED, VFAP-IMMED  , XCP-YES
                           LOG-YES, RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-RET , RCSP-RET  _
        CSMP0097I 13.47.55 CPU-B SS-BSS  SSU-HPN  IS-01
        VFAC0011I 13.47.55 FLUSH REQUEST COMPLETED FOR RECORD ID C1F1,
        NUMBER RECORDS FLUSHED          0+
                           RTP0-4DPA, RTP1-4DPA, RTP2-NIU , RTP3-NIU , RTP4-NIU ,
                           RTP5-NIU , RTP6-NIU , RTP7-NIU , RTP8-NIU , RTP9-NIU ,
          * - VFA IN STAGED MODE _
        RTDM0010I 13.47.55 RECID-A1  , VFAF-SIMMED , VFAP-IMMED  , XCP-YES
                           LOG-YES, RESTORE-YES, UEXIT-NO , LOCKF-DASD, LOCKP-DASD
                           RCSF-RET , RCSP-RET  _
                           RTP0-4DPA, RTP1-4DPA, RTP2-NIU , RTP3-NIU , RTP4-NIU ,
                           RTP5-NIU , RTP6-NIU , RTP7-NIU , RTP8-NIU , RTP9-NIU ,

        RTDM0004I 13.47.55 MODIFY  PROCESSING COMPLETED FOR IMAGE TPF02   +
```

Related Information

- See *TPF System Generation* for a more detailed description of the RIAT values.
- See *TPF System Installation Support Reference* or *TPF System Generation* for more information about the RIAT.
- See *TPF General Macros* for more information about the GETFC macro.

ZRTDM RESET–Reset File RIAT Control Value

Use this command to reset the record ID attribute table (RIAT) control value in the RIAT control record.

Requirements and Restrictions

None.

Format

```
►►──ZRTDM RESET── ──resetval──────────────────────────────────────────►◄
```

resetval
is an 8-digit hexadecimal control value in the RIAT control record.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZRTDM HELP
  ZRTDM ?
- If you do not know the control value to specify, enter the ZRTDM DISPLAY command with the CTLVAL parameter specified to view the RIAT control value on file. Then, enter the ZRTDM RESET command specifying that control value.

Examples

The following example resets the RIAT control value that is on file.

```
User:   ZRTDM RESET AF18764F

System: RTDM0005I 15.55.45 ZRTDM RESET COMPLETED SUCCESSFULLY
```

Related Information

- See “ZRTDM DISPLAY–Display RIAT Entries” on page 1240 for more information about the ZRTDM DISPLAY command and the CTLVAL parameter.
- See *TPF System Generation* for a more detailed description of the RIAT values.
- See *TPF System Installation Support Reference* or *TPF System Generation* for more information about the RIAT.

ZSDEA–Deactivate Pool Addresses

Use this command to deactivate pool addresses that are not in use, from a pool section. Doing this prevents the TPF system from using these pool addresses, so the pool section can be reallocated when all pool addresses are deactivated.

Requirements and Restrictions

None.

Format

```
►►──ZSDEA── ──┬─TEST─────┬────────────────────────────────────────────────►◄
              ├─FILE─────┤
              └─FALLBACK─┘
```

TEST
runs all the error checking for the deactivation function for file pool directory records (FPDRs).

FILE
moves the deactivated FPDRs from the pool rollin directory (#SONRI) to the recoup deactivation history directory (#SONDE).

FALLBACK
moves the deactivated FPDRs from #SONDE to #SONRI.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZSDEA HELP
  ZSDEA ?
- A FILE request followed by another FILE request generates an error message.

Examples

The following example tests the new pool configuration.

```
User:   ZSDEA TEST

System: B1A50001I 15.53.24 PROCESSING ITEM 122
                           ORDINALS      21 THRU      21
        B1A50004I 15.53.24 END OF POOL SECTION SLT
                           RETURN CNT       1650
        B1A50001I 15.53.24 PROCESSING ITEM 123
                           ORDINALS     153 THRU     157
        B1A50005I 15.53.24 EOJ - POOL DEACTIVATION
        B1A50004I 15.53.24 END OF POOL SECTION 4LT
                           RETURN CNT      36000
        B1A90002I 15.53.24 ONLINE DEACTIVATION RECORD GENERATED
        B1A90001I 15.53.24 PROCESSING INTERVAL 122
                           ORDINALS      21 THRU      21
        B1A90001I 15.53.24 PROCESSING INTERVAL 123
                           ORDINALS     153 THRU     157
```

The following example moves the deactivated FPDRs.

```
User:   ZSDEA FILE

System: B1A50001I 15.57.05 PROCESSING ITEM 122
                           ORDINALS      21 THRU      21
        B1A50004I 15.57.05 END OF POOL SECTION SLT
                           RETURN CNT       1650
        B1A50001I 15.57.06 PROCESSING ITEM 123
                           ORDINALS     153 THRU     157
        B1A50004I 15.57.06 END OF POOL SECTION 4LT
                           RETURN CNT      36000
        B1A50005I 15.57.06 EOJ - POOL DEACTIVATION
        B1A90002I 15.57.06 ONLINE DEACTIVATION RECORD GENERATED
        B1A90001I 15.57.06 PROCESSING INTERVAL 122
                           ORDINALS      21 THRU      21
        B1A90001I 15.57.06 PROCESSING INTERVAL 123
                           ORDINALS     153 THRU     157
```

The following example moves the deactivated FPDRs from the pool deactivation directory (#SONDE) to the pool rollin directory (#SONRI).

```
User:   ZSDEA FALLBACK

System: B1A50001I 17.01.14 PROCESSING ITEM 122
                           ORDINALS      21 THRU      21
        B1A50004I 17.01.15 END OF POOL SECTION SLT
                           RETURN CNT       1650
        B1A50001I 17.01.15 PROCESSING ITEM 123
                           ORDINALS     153 THRU     157
        B1A50004I 17.01.15 END OF POOL SECTION 4LT
                           RETURN CNT      36000
        B1A50005I 17.01.15 EOJ - POOL DEACTIVATION
        B1A90002I 17.01.15 ONLINE DEACTIVATION RECORD GENERATED
        B1A90001I 17.01.15 PROCESSING INTERVAL 122
                           ORDINALS      21 THRU      21
        B1A90001I 17.01.15 PROCESSING INTERVAL 123
                           ORDINALS     153 THRU     157
```

Related Information

See the pool generation and reallocation procedures in *TPF Database Reference* for more information about deactivating pool addresses.

ZSELD–Dump Selected Records

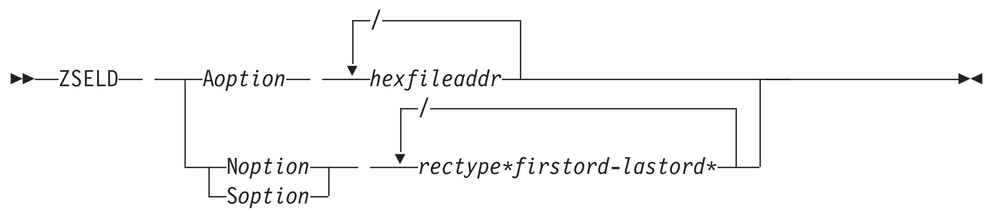
Use this command to start the selective file dump (SFD) debugging tool. The SFD debugging tool writes the contents of specified file records to the real-time tape.

Use the SFD debugging tool and the selective file trace (SFT) debugging tool (also referred to as the selective file dump and trace (SFDT) debugging tools) to locate file-related errors during online operations or while testing under the control of the program test vehicle (PTV) utility.

Requirements and Restrictions

None.

Format



A indicates that the file address is expressed as a file address reference format (FARF) record type. Chained records will contain a FARF file address in the forward chain field.

N indicates that the file address is expressed as a file address compute (FACE) program record type and ordinal number. Chained records will contain a FACE-type ordinal in the forward chain field.

S indicates that the file address is a FACE record type and ordinal number. Chained records will contain a FARF file address in the forward chain field.

option
is one of the following dump options:

2 dumps chained records that have a standard header for 4-byte file addresses.

3 does not dump chained records.

8 dumps chained records that have a standard header for 8-byte file addresses.

hexfileaddr
is the file address of the record specified as an 8- or 16-digit hexadecimal value. You can specify up to 82 file addresses.

*rectype******firstord***-***lastord******
is the file address specified as one or more sets of records. Each set consists of a series of file records of one FACE record type with contiguous FACE ordinal numbers, specified with a 19- or 39-byte field, where:

rectype
is the FACE record type indicator in hexadecimal characters.

firstord
is the 6- or 16-digit hexadecimal ordinal number of the first record in the set.

lastord
is the 6- or 16-digit hexadecimal ordinal number of the last record in the set.

You can specify up to 43 fields with one command. The number of records specified in a single command cannot exceed 420.

Additional Information

All output from the SFDT debugging tools is written to the real-time tape (RTL/RTA). SFD and SFT output may be interspersed with other test or control program output also on the real-time tape. Use the offline diagnostic output formatter (DOF) utility to process the SFDT output.

Examples

The following example writes the contents of the specified file record to the real-time tape.

```
User:   ZSELD A2 184800B3

System: SFD COMPLETE
```

Related Information

- See “ZTRCE–Trace Selected Records” on page 1387 and “ZTHLT–Stop Tracing Selected Records” on page 1342 for more information about starting and stopping the SFT debugging tool.
- See *TPF Program Development Support Reference* for more information about the SFDT debugging tools and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline DOF utility.
- See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about the PTV utility.

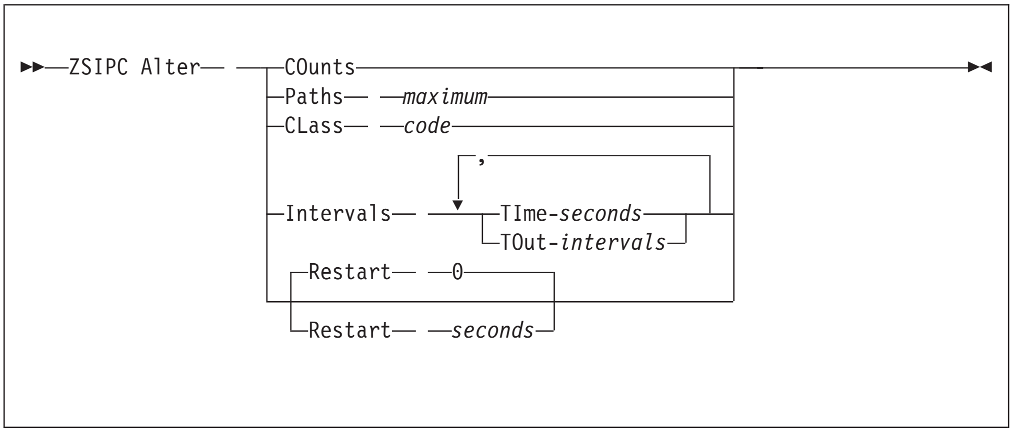
ZSIPC ALTER–Alter System Interprocessor Information for MPIF/IPC

Use this command to change the interprocessor communications (IPC) control information, which includes counters, paths, the path class, and intervals.

Requirements and Restrictions

This command is valid only for Multi-Processor Interconnect Facility (MPIF) IPC users.

Format



COunts
resets all IPC counters to 0.

Paths *maximum*
sets the maximum number of paths between processors, where *maximum* is a 1- to 2-digit decimal value.

Note: The maximum number of paths takes effect when the next initial program load (IPL) is performed.

CLass *code*
sets the path class that will be used by IPC, where *code* is the 1-character alphanumeric path class code that identifies the path class.

Note: The new class designation takes effect when the next IPL is performed.

Intervals
modifies the current interval parameters or specifies new interval parameters.

TIme-*seconds*
is the 1- to 2-digit number of seconds for the IPC interval.

Note: This value takes effect when the next IPL is performed.

TOut-*intervals*
is the 1- to 2-digit number of intervals that IPC waits for connections to be established to a destination processor before declaring a transmission failure. IPC returns the request to the sender.

Note: This value takes effect when the next IPL is performed.

Restart *seconds*
is the number of seconds from 0–65 535, that IPC waits for connections to be established to all loosely coupled processors during restart. A 0 value indicates an infinite wait.

Note: This value takes effect when the next IPL is performed.

Attention: Using this parameter could compromise the integrity of the loosely coupled complex.

Additional Information

Enter the ZSIPC DISPLAY command to display information about MPIF IPC.

Examples

All the IPC counters are set to 0 in the following example.

```
User:   ZSIPC ALTER COUNTS

System: SIPC0002I 13:22:21 COUNTS ZEROED
```

The maximum number of paths between processors is set to 10 in the following example.

```
User:   ZSIPC ALTER PATHS 10

System: SIPC0006I 12:23:44 MAXIMUM NUMBER OF PATHS BETWEEN PROCESSORS ALTERED TO 10
```

The path class used by IPC is set to C in the following example.

```
User:   ZSIPC ALTER CLASS C

System: SIPC0007I 14:21:33 IPC CLASS ALTERED TO C
```

The IPC interval is set to 10 and the number of intervals that IPC waits for connections to be established is set to 40 in the following example.

```
User:   ZSIPC ALTER INTERVALS TIME-10,TOUT-40

System: SIPC0005I 11:42:57 INTERVALS SET
```

The number of IPC intervals that IPC waits for connections to be established is set to 30 in the following example.

```
User:   ZSIPC ALTER INTERVALS TOUT-30

System: SIPC0005I 11:42:57 INTERVALS SET
```

The number of seconds that IPC waits for connections to be established to all loosely coupled processors during restart is set to 240 in the following example.

```
User:   ZSIPC ALTER RESTART 240

System: SIPC0008I 16:32:46 IPC RESTART TIMEOUT VALUE ALTERED TO 240
```

Related Information

See *TPF Main Supervisor Reference* for more information about interprocessor communications.

ZSIPC DISPLAY–Display System Interprocessor Information for MPIF/IPC

Use this command to display information about Multi-Processor Interconnect Facility (MPIF) interprocessor communications (IPC).

Requirements and Restrictions

This command is valid only for MPIF IPC users.

Format

```
►►──ZSIPC Display── ──┬─Status──────────────────────────┬──────────────────────►◄
                      └─COunts─┬─────────────────────┬──┘
                               └─ ──ID-─┬─cpuid─┬────┘
                                        └─ALL───┘
```

Status
displays the maximum number of paths between processors, the path class used by IPC, the restart connection timeout value, and the amount of time IPC waits for a connection to be established before declaring a connection failure.

COunts
displays the send and receive counts for the processors in a loosely coupled complex.

ID specifies the processor.

cpuid
is the 1-character alphanumeric CPU ID of a processor.

ALL
displays the send and receive counts for all the processors.

Additional Information

Enter the ZSIPC ALTER command to change the IPC control information.

Examples

The status for MPIF IPC is displayed in the following example.

```
User:   ZSIPC DISPLAY STATUS

System: SIPC0000I 14.28.05 SIPC STATUS
         TIME IN SECONDS - 0010
         INTERVALS TIME-01 TOUT-10
         RESTART TIMEOUT INTERVAL -     0
         MAXIMUM NUMBER OF PATHS - 10
         IPC CLASS - A
```

The send and receive counts for CPU B are displayed in the following example.

```
User:   ZSIPC DISPLAY COUNTS ID-B

System: SIPC0001I 14.28.52 TOTALS FOR HOST-B
        XMITD-00000000  RECVD-00000000
        RTRND-00000000  ERRTN-00000000
        SCRPB-00000000
```

Related Information

See *TPF Main Supervisor Reference* for more information about interprocessor communications.

ZSLDR–Activate Data Loader

Use this command to load fixed file records from a pilot tape to the TPF database.

Requirements and Restrictions

You can enter this command only when the TPF system is in 1052 state unless the ID of the pilot tape is N. If the ID of the pilot tape is N, the TPF system can be in any state.

Format

```
►►—ZSLDR— —LOAD— —DATA— —pilotid————————————————————————————►◄
```

pilotid
is the ID of the pilot tape created by the system test compiler (STC) program.

Additional Information

Once records are loaded to the TPF database, no utility is available to remove the records from the database. If it is necessary to delete previously loaded records from the TPF database, you must create and load a new pilot tape that contains dummy records. If the number of records decreases from one load to the next, the additional records from the first load will ***not*** be removed from the database.

Examples

Fixed file records are loaded from a pilot tape in the following example.

```
User:   ZSLDR LOAD DATA T

System: ACPD0011I 17.32.44 SYSTEM DATA LOAD STARTED

        ACPD-SYSTEM DATA LOAD COMPLETE
                  RECORDS READ = 01678  RECORDS WRITTEN = 01678
```

Related Information

- See *TPF System Installation Support Reference* for more information about the data loader.
- See *TPF Program Development Support Reference* for more information about the STC program used to create pilot tapes.

ZSNDA–Send an Unsolicited System Message to an Application

Use this command to send an unsolicited message to the consoles in session with a specific application. You can also send an unsolicited message to all the consoles in session with any application on the current processor.

Requirements and Restrictions

The TPF system must be in CRAS state or higher.

Format

```
                                    (1)
►►──ZSNDA───┬─<new line key>──────────────────────┬─message──────────────────────►◄
            └──name───┬──────────────────────────┘
                      │                    (1) │
                      └─<new line key>─────────┘
```

Notes:

1 The <new line key> characters indicate that you press the key on the keyboard to move to a new line before continuing with the syntax. Do not type the <new line key> characters.

<new line key>
: is the key on the keyboard that you press to move to a new line.

name
: is the 1- to 4-character alphanumeric name of an application. If you do not specify an application name, the unsolicited message is sent to all the consoles in session with any application on the current processor.

message
: is a 1- to 255-character alphanumeric message.

Additional Information

The consoles that receive the unsolicited message are immediately notified of the pending message. The operators at these consoles must enter the LOGU command to display the unsolicited message.

Examples

In the following example, a message is sent to the consoles in session with the SMPB application.

```
User:   ZSNDA SMPB THIS IS A TEST

System: SNDA0001I 12.43.49 REQUEST PROCESSED
```

In the following example, another message is sent to the consoles in session with the SMPB application. Notice that you can enter the message on a separate line by pressing the key on the keyboard that moves you to a new line.

```
User:   ZSNDA SMPB
        TEST IS ALSO A TEST

System: SNDA0001I 12.51.50 REQUEST PROCESSED
```

In the following example, a message is sent to all consoles in session with any of the applications on the current processor. Notice that you must press the key on the keyboard that moves you to a new line before you type the message.

```
User:   ZSNDA
        BROADCASTING A MESSAGE TO ALL CONSOLES ON THIS PROCESSOR

System: SNDA0001I 12.51.02 REQUEST PROCESSED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the unsolicited message processor.

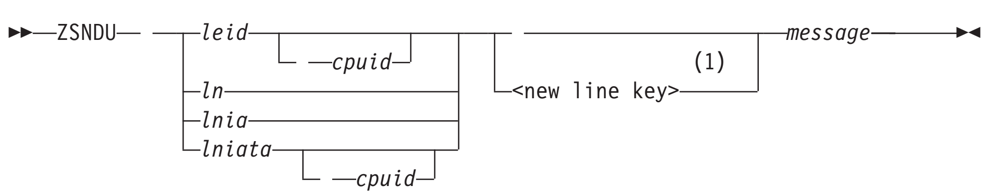
ZSNDU–Send an Unsolicited System Message to a Terminal or Logical Unit

Use this command to send an unsolicited message to 1 or more consoles or SNA logical units.

Requirements and Restrictions

- The TPF system must be in CRAS state or higher.
- Do not enter this command to send unsolicited messages to the prime CRAS console. If you want to send an unsolicited message to the prime CRAS console, enter the ZRCRS command.
- You cannot send an unsolicited message to network extension facility (NEF) lines or interchanges in an Advanced Communication Function (ACF) system with NEF because the TPF system is not aware of the NEF lines or the terminal interchanges.

Format



Notes:

1 The <new line key> characters indicate that you press the key on the keyboard to move to a new line before continuing with the syntax. Do not type the <new line key> characters.

leid
sends an unsolicited message to the console at the specified 4-digit hexadecimal logical end-point identifier.

ln
sends an unsolicited message to the consoles at the specified 2-character hexadecimal line number.

lnia
sends an unsolicited message to the consoles at the specified 4-character hexadecimal line number and interchange address.

lniata
sends an unsolicited message to the console at the specified 6-character hexadecimal line number, interchange address, and terminal address.

cpuid
is the 1-character alphanumeric CPU ID of a processor.

<new line key>
is the key on the keyboard that you press to move to a new line.

message
is a 1- to 255-character alphanumeric message.

Additional Information

The consoles that receive the unsolicited message are immediately notified of the pending message. The operators at these consoles must enter the LOGU command to display the unsolicited message.

Examples

A message is sent to the specified console on processor C in the following example.

```
User:   ZSNDU FE0302 C THIS IS TEST

System: SNDU0001I 13.53.44 REQUEST PROCESSED
```

A message is sent to all the consoles on the specified line in the following example. Notice that you can type the message on a separate line by pressing the key on the keyboard that moves you to a new line.

```
User:   ZSNDU FE
        TEST IS A TEST

System: SNDA0001I 12.51.50 REQUEST PROCESSED
```

Related Information

See *TPF Data Communications Services Reference* for more information about the unsolicited message processor.

ZSNMP–Simple Network Management Protocol

Use this command to do the following:

- Refresh the `/etc/snmp.cfg` Simple Network Management Protocol (SNMP) configuration file and copy it into core storage.
- Retrieve TPF Management Information Base (MIB) variables.

Requirements and Restrictions

- You can enter this command only in CRAS state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZSNMP── ──┬─REFresh──────────────────────────────────────┬──────────►◄
              └─Display── ──┬─mibvar───┬──┬─────────────────┬─┘
                            ├─mibasn───┤  └── ──File-filename─┘
                            └─mibgroup─┘
```

REFresh
refreshes the `/etc/snmp.cfg` SNMP configuration file and copies it into core storage.

Display
retrieves the specified MIB variable from the TPF system.

mibvar
is the MIB variable descriptor. Specify the standard MIB abbreviation of the object name or table. See *TPF Transmission Control Protocol/Internet Protocol* for a list of the MIB variables that are supported by the TPF system and the associated variable descriptors.

mibasn
is the MIB object identifier in Abstract Syntax Notation (ASN.1) format. Specify the entire object identifier up to the group, variable, or index level. See *TPF Transmission Control Protocol/Internet Protocol* for a list of the MIB variables that are supported by the TPF system and the associated object identifiers.

mibgroup
is the MIB group name. Specify one of the following:

SYSTEM
displays the variables in the system group.

INTERFACES
displays the variables in the interfaces group.

IP displays the variables in the IP group.

ICMP
displays the variables in the ICMP group.

TCP
displays the variables in the TCP group.

UDP
displays the variables in the UDP group.

SNMP
displays the variables in the SNMP group.

MIB
displays all of the MIB variables.

UMIB
displays your enterprise-specific MIB variables.

See *TPF Transmission Control Protocol/Internet Protocol* for a list of the MIB variables that are supported by the TPF system and the group to which each variable belongs.

File-*filename*
retrieves the specified MIB variable from the TPF system and sends the output to a file, where *filename* is the name of the file in which you want to save the data.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZSNMP HELP
  ZSNMP ?
- The `/etc/snmp.cfg` SNMP configuration file is automatically refreshed every time the TPF system cycles to CRAS state or above.

Examples

The following example refreshes the `/etc/snmp.cfg` SNMP configuration file and copies it into core storage.

```
User:   ZSNMP REFRESH

System: SNMP0001I 09.17.22 SNMP CONFIGURATION FILE REFRESHED SUCCESSFULLY
```

The following example requests the MIB variable information for SNMP input packets using the variable descriptor.

```
User:   ZSNMP DISPLAY SNMPINPKTS

System: SNMP0018I 09.50.33 SNMP MIB RETRIEVAL DISPLAY
        snmpInPkts : 247
        END OF DISPLAY
```

The following example requests the MIB variable information for SNMP input packets using ASN.1 format.

```
User:   ZSNMP DISPLAY 1.3.6.1.2.1.11.1

System: SNMP0018I 09.50.51 SNMP MIB RETRIEVAL DISPLAY
        snmpInPkts : 247
        END OF DISPLAY
```

The following example requests the MIB variable information for the UDP group.

```
User:   ZSNMP DISPLAY UDP

System: SNMP0018I 10.10.32 SNMP MIB RETRIEVAL DISPLAY
        udpInDatagrams : 9436
        udpNoPorts : 21
        udpInErrors : 0
        udpOutDatagrams : 6
        udpLocalAddress.0.0.0.0.69 : 0.0.0.0
        udpLocalAddress.0.0.0.0.520 : 0.0.0.0
        udpLocalPort.0.0.0.0.69 : 69
        udpLocalPort.0.0.0.0.520 : 520
        END OF DISPLAY
```

The following example requests the MIB variable information for the UDP group and saves the data to a file.

```
User:   ZSNMP DISPLAY UDP FILE-udpgroup.txt

System: SNMP0019I 09.51.28 SNMP MIB RETRIEVAL CONTENTS SENT TO FILE udpgroup.txt
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about SNMP.

ZSOCK–TCP/IP Network Tools

Use this command to do the following:

- Display TCP/IP native stack support control block information
- Display the number of bytes sent and received across an individual TCP/IP socket in a 5-second interval
- Convert TCP/IP native stack support resource information
- Deactivate TCP/IP native stack support sockets
- Display a summary table of socket descriptors and selected socket control block information.

Requirements and Restrictions

- You can enter the ZSOCK command only in 1052 state or higher.
- You can enter the ZSOCK command only from the basic subsystem (BSS).
- The ZSOCK command is only supported for sockets and Internet Protocol (IP) addresses using TCP/IP native stack support.

Format

```
►►──ZSOCK── ──┬─DISPlay──┬──────────┬──┬─┤ Additional Parameters ├─┬──┬──────►◄
              │          └─ ──FORMAT─┘  └─ ──SOCKet-socknum─────────┘  │
              ├─┬─CONVert─┬──┬─┤ Additional Parameters ├─┬────────────┤
              │ └─INACT───┘  └─ ──SOCKet-socknum──────────┘            │
              ├─SUMmary──┤ Additional Parameters ├─────────────────────┤
              └─DATAFLOW── ──SOCKet-socknum───────────────────────────┘

Additional Parameters:

├──┬───────────────┬──┬───────────────┬──┬───────────────┬──────────────────►
   └─ ──LIP-lipaddr─┘  └─ ──LPORT-lport─┘  └─ ──RIP-ripaddr─┘

►──┬───────────────┬──┬─────────────────┬────────────────────────────────────┤
   └─ ──RPORT-rport─┘  └─ ──PROTOcol-prot─┘
```

DISPlay
: displays TCP/IP native stack support control block information.

FORMAT
: formats the requested information for display.

LIP-*lipaddr*
: specifies a local IP address, where *lipaddr* is the numeric IP address. Do not enter the following IP addresses for this parameter:
  - 0.0.0.0
  - 255.255.255.255

LPORT-*lport*
: specifies a local port number, where *lport* is a decimal number from 1 to 65 535.

RIP-*ripaddr*
specifies a remote IP address, where *ripaddr* is the numeric IP address of the remote server. Do not enter the following IP addresses for this parameter:
- 0.0.0.0
- 255.255.255.255

RPORT-*rport*
specifies a remote port number, where *rport* is a decimal number from 1 to 65 535.

PROTOcol-*prot*
specifies the protocol of a socket descriptor, where *prot* is one of the following protocols:

ICMP
Internet Control Message Protocol

IP Internet Protocol

RAW
Raw Internet Protocol

TCP
Transmission Control Protocol

UDP
User Datagram Protocol

SOCKet-*socknum*
specifies a socket descriptor, where *socknum* is the 6- to 8-digit hexadecimal socket descriptor.

CONVert
finds the specified socket descriptor or the corresponding socket descriptor that matches the selection criteria and displays the socket descriptor with the socket control block values for all the possible selection criteria.

INACT
deactivates a specific socket or all sockets that have matching values for all the specified selection criteria.

SUMmary
displays a summary table of all socket descriptors matching the specified selection criteria.

DATAFLOW
displays the number of bytes sent and received across an individual TCP/IP socket in a 5-second interval. The DATAFLOW parameter is used only for User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) sockets.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZSOCK HELP
ZSOCK ?

Examples

The following example displays formatted socket information based on a specified socket descriptor.

```
User:   ZSOCK DISP FORMAT SOCK-C0001F

System: SOCK0010I 10.45.54 SOCKET CONTENTS FORMATTED
        LOCAL IP  -    9.117.236.131 LOCAL PORT  -   1025
        REMOTE IP -    9.117.241.140 REMOTE PORT -   9999
        PROTOCOL - TCP               SOCKET TYPE - STREAM
        SOCKET DESCRIPTOR - 00C0001F _
        NON BLOCKING MODE - N        SEND WINDOW BLOCKED - N
        AOR PENDING - N              CLOSE ISSUED - N
        SEND BUFF SIZE -     131072  SEND BUFF IN USE -        100
        RECV BUFF SIZE -     131072  RECV BUFF IN USE -          0
        NEXT SEND SEQ  - 2821439565  LAST ACKED SEQ   - 2821439465
        NEXT RECV SEQ  - 2821570925  MAX PACKET SIZE  -       1492
        WINDOW SCALE   -          4  SEND WINDOW SIZE -     131072
        STATE - ESTABLISHED          AVG ROUND TRIP       0.273818
        FIRST HOP IP   - 9.117.241.140
        END OF DISPLAY
```

The following example displays unformatted socket information based on a specified socket descriptor.

```
User:   ZSOCK DISPLAY SOCKET-c0001F

System: SOCK0011I 10.46.17 SOCKET CONTENTS

        0BD9B238- 00000000 00000000 00000000 00C0001F ........ ........
        0BD9B248- 00000000 00000000 00000000 0975EC83 ........ .......c
        0BD9B258- 00000000 00000000 00000000 0975F18C ........ ......1.
        0BD9B268- 0401270F 01060400 0BB140D0 0BB140D0 ........ ........
        0BD9B278- 00020000 00000000 00000064 00000000 ........ ........
        0BD9B288- 00000000 00000000 00020000 00000000 ........ ........
        0BD9B2A8- 00000000 00000000 02009000 00000000 ........ ........
        0BD9B2B8- 00000000 FF000000 FF000000 00000000 ........ ........
        0BD9B2D8- 00000064 00000000 00000000 00000000 ........ ........
        0BD9B2E8- 06000000 00000000 00000000 00000000 ........ ........
        0BD9B2F8- 00000000 00000000 0400001E 05D40000 ........ .....M..
        0BD9B308- A82BDC6D A82BDC09 00020000 A82DDD8D y.._y... ....y...
        0BD9B318- 00000000 00000000 00000000 00004004 ........ ...... .
        0BD9B328- 00000000 00000000 00000000 0BD9B35C ........ .....R.*
        0BD9B338- 00049920 00000000 00000000 00000000 ..r..... .......c
        0BD9B358- 0975F18C                            ..1.
        END OF DISPLAY - ZEROED LINES NOT DISPLAYED
```

The following example displays formatted socket information based on a specified local IP address, remote IP address, local port and remote port.

```
User:   ZSOCK DISP FORMAT LIP-9.117.241.12 RIP-9.117.241.140 LPORT-1031 RPORT-9999

System: SOCK0010I 10.47.15 SOCKET CONTENTS FORMATTED
        LOCAL IP  -    9.117.236.130 LOCAL PORT  -    9999
        REMOTE IP -    9.117.241.140 REMOTE PORT -    1025
        PROTOCOL - TCP               SOCKET TYPE - STREAM
        SOCKET DESCRIPTOR - 00C00021 _
        NON BLOCKING MODE - N        SEND WINDOW BLOCKED - N
        AOR PENDING - N              CLOSE ISSUED - N
        SEND BUFF SIZE -     131072  SEND BUFF IN USE -        100
        RECV BUFF SIZE -     131072  RECV BUFF IN USE -          0
        NEXT SEND SEQ  - 2843729236  LAST ACKED SEQ   - 2843729136 _
        NEXT RECV SEQ  - 2843569842  MAX PACKET SIZE  -       1492
        WINDOW SCALE   -          4  SEND WINDOW SIZE -     131072
        STATE - ESTABLISHED          AVG ROUND TRIP  -  0.182410
        FIRST HOP IP -   9.117.241.140
        END OF DISPLAY
```

The following example converts socket information based on a specified local IP address, remote IP address, local port and remote port.

```
User:   ZSOCK CONV LIP-9.117.236.130 RIP-9.117.241.140 LPORT-9999 RPORT-1065

System: SOCK0019I 10.47.45 BEGIN CONVERT DISPLAY
        LOCAL IP  -    9.117.236.130   LOCAL PORT  -   9999
        REMOTE IP -    9.117.241.140   REMOTE PORT -   1025
        SOCKET DESCRIPTOR - 00C00021
        PROTOCOL - TCP
        END OF DISPLAY
```

The following example displays a summary report of sockets based on a specified protocol.

```
User:   ZSOCK SUM PROTO-TCP

System: SOCK0021I 10.45.14 SOCKET SUMMARY INFORMATION
        SOCKET     LOCAL          LOCAL   REMOTE        REMOTE    PROT   STATE
         DESC       IP             PORT     IP           PORT

        00C0001E                   9999                            TCP    LISTEN
        00C0001F   9.117.236.131   1025   9.117.241.140   9999     TCP    ESTABLI
        00C00020   9.117.241.139   9999   9.117.241.140   1024     TCP    ESTABLI
        00C00021   9.117.236.130   9999   9.117.241.140   1025     TCP    ESTABLI
        SUMMARY TOTAL        4
        END OF DISPLAY
```

The following example deactivates sockets based on a specific remote port.

```
User:   ZSOCK INACT RPORT-1025

System: SOCK0018I 10.48.18 INACT COMPLETE,    1 SOCKETS CLOSED+
```

The following example displays the data flow statistics of a socket descriptor.

```
User:   ZSOCK DATAFLOW SOCK-C00031

System: SOCK0024I 10.47.34 BEGIN PROCESSING SOCKET DATAFLOW STATISTICS
        CSMP0097I 10.47.39 CPU-B SS-BSS SSU-HPN IS-01
        SOCK0025I 10.47.39 SOCKET DATAFLOW STATISTICS FOR A 5-SECOND INTERVAL
        SOCKET DESCRIPTOR-C00031
               BYTES SENT      - 23430
               BYTES RECEIVED  -     0
        END OF ZSOCK DATAFLOW SOCKET DISPLAY
```

Related Information

None.

ZSONS ALTER ERROR COUNTS–Alter Disk Error Counts

Use this command to set the correctable and uncorrectable counts of disk errors to 0.

Requirements and Restrictions

None.

Format

```
►►──ZSONS Alter Error Counts── ──┬─module──┬──────────┬──┬──────────┬─┬──────►◄
                                 │         └─ ─number─┘  └─ ─Online─┘ │
                                 └─ALL────────────────────────────────┘
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time packs or a pseudo module number for general files.

number
: clears a specific number of modules beginning with the specified module, where *number* is the number of modules from X'01'–X'FFF'.

Online
: clears the error counts for all the online modules up to the specified number of modules or the maximum limit.

ALL
: clears the error counts for all modules.

Additional Information

- If you do not specify the number of modules to clear or the ONLINE parameter, all the online modules beginning at the starting module and ending at the predefined maximum limit are cleared.

  Note: The maximum number of modules that can be cleared is originally set to X'18' by the TPF system.
- Enter the ZSONS DISPLAY ERROR COUNTS command to display the current correctable and uncorrectable counts of disk errors.

Examples

The correctable and uncorrectable counts of disk errors are set to 0 for all the modules in the following example.

```
User:   ZSONS A E C ALL

System: SONS0005I 09.40.59 ALL ERROR COUNTERS CLEARED
```

The correctable and uncorrectable counts of disk errors are set to 0 for the 047 and 048 module in the following example, where:

MOD
: is the symbolic or pseudo module number.

TYPE
is the DASD device model number and one of the following indicators:

ECKD
Module is running in extended count key data mode.

C/D
Module is running in count key data mode on a caching control unit.

C/E
Module is running in extended count key data mode on a caching control unit.

RCS
Module is running in extended count key data mode on a full function record caching control unit.

Note: NMT indicates that the module is not mounted.

SDA
is the symbolic device address.

USE
is the way the module is being used. The possible uses are:

GEN
Module is a pseudo general file.

RLT
Module is a real-time file.

GDS
Module is a general data set.

DUP
is the symbolic module number of the duplicate module, if one exists.

STAT
is the module status. The possible conditions are:

ON
Module is online.

OFF
Module is offline.

COPY
Module is being copied ***from*** by ALL file copy or is being copied ***to*** by duplicate file update.

VSN
is the volume serial number.

LOCK
is the lock status. The possible conditions are:

CFLF
Module is running on a concurrency filter lock facility.

LLF
Module is running on a limited lock facility.

NONE
Module currently has no locking.

COR
: is the correctable error count (in decimal).

UNC
: is the uncorrectable error count (in decimal).

```
User:   ZSONS A E C 047 2

System: SONS0004I 09.40.59
              MOD TYPE     SDA  USE DUP STAT  VSN   LOCK COR   UNC
              047 3380     0EE5 RLT 048   ON BP0001 NONE 00000 00000 CLEARED
              048 3380     0463 RLT 047   ON BP0002 NONE 00000 00000 CLEARED
        DISPLAY COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS ALTER ERROR LIMITS–Alter Error Limits

Use this command to set the number of errors and the time interval that is used by the DASD error recovery routine to determine when a module should be taken offline because of excessive errors.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Alter Error Limits── ──Count-number── ──Interval-seconds──────────►◄
```

Count-*number*
: specifies the error count value, where *number* is a decimal number from 0–9999.

Interval-*seconds*
: specifies the length of the time interval, where *seconds* is the number of seconds from 0–9999.

Additional Information

- The error limits are checked only when both the COUNT parameter and the INTERVAL parameter are greater than 0.
- The DASD error recovery routine counts the number of errors received in a specified time period. If this count exceeds the specified limit, the DASD error recovery routine sets a flag and repeats the same logic. If the limit is exceeded again, the module is taken offline. During the first interval, only the first error is reported. During the second interval, error messages are reported in order to acquaint you with the types of errors that are occurring. If the interval ends before the count threshold is reached, everything is reset and limit checking starts again.

  There are two reasons for the logic. The first is to try to limit the number of messages sent to the console in order to prevent low core conditions or console flooding. The second is to provide a mechanism by which the DASD error recovery routine can remove modules that are receiving an excessive number of errors.
- Enter the ZSONS DISPLAY ERROR LIMITS command to display the current error limit count and time interval.

Examples

A module is taken offline if it receives 10 errors in 2 consecutive 5-second intervals in the following example.

```
User:   ZSONS A E L C-10 I-5

System: SONS0003I 09.40.59 ERROR LIMITS COUNT-  10, INTERVAL-   5
```

In the following example, the error limits are not checked.

ZSONS ALTER ERROR LIMITS

```
User:   ZSONS A E LIM C-0 I-0

System: SONS0003I 09.40.59 ERROR LIMITS COUNT-   0, INTERVAL-   0
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS ALTER HALT–Alter Halt Timeout Value

Use this command to set the timeout value for the lost interrupt recovery routine in the DASD missing interrupt handler (DMIH). The DMIH monitors the DASD I/O to ensure that the DASD devices are responding in a timely method.

This command does the following:

1. Starts the DMIH (if the value specified for this command is not 0).
2. The DMIH attempts to stop and restart requests that were active for longer than a specified period of time and, as a result, issues a CTL-03A dump.
3. If the request does not complete again in the specified time interval, the DMIH stops the request and takes the module offline.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Alter Halt── ──Time-seconds──────────────────────────────────────►◄
```

Time-*seconds*
: specifies the time interval that the DMIH waits before stopping and restarting a request after a 5-second timeout interval ends, where *seconds* is the number of seconds from 0–999.

Notes:

1. If you specify 0, no recovery routines are started.
2. The time interval is initially set to 0 by the TPF system.

Additional Information

- When running on a VM system, the 5-second timeout interval and the value specified for this command are doubled.
- The CYED program also uses the value specified for this command to recover from stalled module queues as follows:
  1. Determines that the module queue is stalled and the normal DMIH recovery is not going to recover from it.
  2. Starts a recovery routine, which issues a CTL-003A dump.
  3. Resets the device and restarts the first request on the module queue.

  Note: The CYED program determines when to start the recovery routine by taking the normal DMIH timeout value (5 seconds), doubling this value, and adding the value specified for this command to it.
- If you specify a value greater than 0 for TPF systems that use nonbuffered or noncache DASD control units, damaged records on the DASD surface can result.
- Enter the ZSONS DISPLAY HALT command to display the current timeout value for the DMIH.

Examples

In the following example, the lost interrupt timeout value for the DMIH recovery routine is set to 15 seconds.

```
User:   ZSONS A H T-15

System: SONS0001I 09.40.59 HALT TIMEOUT VALUE IS  15
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS ALTER SCAN–Alter Scan Time Value

Use this command to specify how often the DASD queues are scanned by the CYED program for I/O requests that are not being processed by the TPF system in a timely method.

Requirements and Restrictions

None.

Format

```
►►──ZSONS Alter Scan── ──Time-seconds──────────────────────────────────►◄
```

Time-*seconds*
indicates the time interval for scanning the module queues, where *seconds* is the number of seconds from 0–999.

Note: The time interval is initially set to 0 by the TPF system.

Additional Information

- When running on a VM system, the specified scan interval is doubled.
- If the value for the ZSONS ALTER HALT command is not set to 0, the CYED program takes the normal DMIH timeout value (5 seconds), doubles that value and adds the halt value to the result. If scan indicates that the queue is stalled for at least that long, the CYED program starts the DMIH recovery routine, which issues a CTL-003A dump, resets the device, and redrives the first request on the module queue.
- The CYED program scans every DASD module queue for outstanding requests. When a queue with an outstanding request is found, the top request on that queue is flagged. When the next scan is performed, the module queue is checked to see if the indicator is still set. If it is, this indicates that the DASD support did not process the request in the specified time period. Therefore, the requests for this module are not proceeding at a reasonable rate. When this occurs, the CYED program sends an attention message and takes no additional recovery action. In summary, if a module queue is actually stalled (or is not completing), the CYED program sends an attention message at each scan interval until the queue does proceed.
- Although the time interval is initially set to 0 by the TPF system, the module queues are actually scanned every 5 seconds by default.
- Enter the ZSONS DISPLAY SCAN command to display the current time interval for scanning the module queues.

Examples

In the following example, the time value used to schedule the next module queue scan is set to 15 seconds.

```
User:   ZSONS A S T-15

System: SONS0002I 09.40.59 SCAN TIME VALUE IS  15
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS ALTER SCP–Alter RCS State Change Pending Timeout Value

Use this command to set the timeout value for the record cache subsystem (RCS) state change pending monitor (SCPM) routine. The RCS SCPM monitors the DASD I/O to ensure that the DASD devices are responding in a timely method. This time value allows the monitoring of RCS devices that have entered a state change pending condition because of an error recovery action at the device or control unit.

This command does the following:

1. Starts the RCS SCPM (if the value specified for this command is not 0).
2. The RCS SCPM attempts to monitor and report stalled module queues because of RCS state change pending conditions for DASD devices.
3. If the state change pending condition persists for a time period exceeding the specified interval, the RCS SCPM will initiate action to take the module offline.

Note: The state change pending condition timing period represents the time period from the point at which the state change pending unit check is reported by the device until a valid ending device status is received for a given I/O operation.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Alter SCP── ──Time-seconds──────────────────────────────────────►◄
```

Time-*seconds*
specifies the time interval that the RCS SCPM waits before initiating action to take the device offline once an RCS state change pending condition is detected, where *seconds* is the number of seconds from 0–9999.

Notes:

1. If you specify 0, no SCPM monitoring will be performed.
2. The time interval is initially set to 0 by the TPF system.

Additional Information

- When running on a VM system, the value specified for this command is doubled.
- Enter the ZSONS DISPLAY SCP command to display the current timeout value for the RCS SCPM.
- The actual amount of time that the state change pending condition persists before the SCPM begins to take the device offline is determined by a combination of the value specified for the TIME parameter and the scan time frequency interval at which the SCPM runs (the default value is 5 seconds). For example, if the RCS timeout value is specified as 15 seconds and the scan time frequency interval is 5 seconds, the following timing events occur:

T0 The device enters a state change pending condition as a result of unit check error presentation.

T1 Timing of the state change pending condition begins at the next scan time interval following T0 and is indicated by the CYED0001I online message. The elapsed time from the recognition of the state change pending condition (T0) until SCPM timing actually begins is based on the scan time frequency. Therefore, in this example, the actual elapsed time from T0 to T1 can be 0 to 5 seconds.

T2 The next scan determines that the state change pending timing is active and reports that the condition still exists through the CYED0004I online message. The actual elapsed time from T0 to T2 can be 5 to 10 seconds.

T3 The next scan determines that the state change pending timing is active and reports that the condition still exists through the CYED0005I online message. The actual elapsed time from T0 to T3 can be 10 to 15 seconds.

T4 The next scan determines that the state change pending timing is still active. The recovery action to take the device offline is now initiated because the RCS timeout value (15 seconds in this example) has elapsed since T1. The actual elapsed time from T0 to T4 can be 15 to 20 seconds.

The previous example showed how the actual state change pending condition could have been active from 15 to 20 seconds before the recovery action to take the device offline begins

Examples

In the following example, the state change pending timeout value for the RCS SCPM recovery routine is set to 15 seconds.

```
User:   ZSONS A SCP T-15

System: SONS0007I 12:22:42 RCS STATE-CHANGE-PENDING TIMEOUT VALUE IS 15
```

Related Information

- See *TPF Database Reference* for more information about DASD support.
- See *Messages (Online)* for more information about the CYED0001I, CYED0004I, and CYED0005I online messages.

ZSONS DISPLAY ERROR COUNTS–Display Disk Error Counts

Use this command to display the correctable and uncorrectable counts of disk errors.

Requirements and Restrictions

None.

Format

```
►►──ZSONS Display Error Counts── ──module──┬──────────┬──┬──────────┬──────────►
                                           └─ ──number─┘  └─ ──Online─┘

►──┬───────┬─────────────────────────────────────────────────────────────────►◄
   └─ ──ALL─┘
```

module
: is a 1- to 3-digit hexadecimal symbolic input module number for real-time packs or the pseudo module number for general files.

number
: displays the error counts for a specific number of modules beginning with the specified module, where *number* is the number of modules from X'01'–X'FFF'.

Online
: displays the error counts for all the online modules up to the specified number of modules or the maximum limit.

ALL
: displays the error counts even if the counts are 0. If you do not specify the ALL parameter, the error counts are not displayed for a device if they are 0.

Additional Information

- If you do not specify the number of modules to display or the ONLINE parameter, the error counts for all the online modules beginning at the starting module and ending at the predefined maximum limit are displayed.

  Note: The maximum number of modules that can be displayed is originally set to X'18' by the TPF system.
- Enter the ZSONS ALTER ERROR COUNTS command to set the correctable and uncorrectable error counts of disk errors to 0.

Examples

The correctable and uncorrectable error counts are displayed in the following example, where:

MOD
: is the symbolic or pseudo module number.

TYPE
: is the DASD device module number and one of the following indicators:

ECKD
Module is running in extended count key data mode.

C/D
Module is running in count key data mode on a caching control unit.

C/E
Module is running in extended count key data mode on a caching control unit.

RCS
Module is running in extended count key data mode on a full function record caching control unit.

Note: NMT indicates that the module is not mounted.

SDA
is the symbolic device address.

USE
is the way the module is being used. The possible uses are:

GEN
Module is a pseudo general file.

RLT
Module is a real-time file.

GDS
Module is a general data set.

DUP
is the symbolic module number of the duplicate module, if one exists.

STAT
is the module status. The possible conditions are:

ON
Module is online.

OFF
Module is offline.

COPY
Module is being copied ***from*** by ALL file copy or is being copied ***to*** by duplicate file update.

VSN
is the volume serial number.

LOCK
is the lock status. The possible conditions are:

CFLF
Module is running on a concurrency filter lock facility.

LLF
Module is running on a limited lock facility.

NONE
Module currently has no locking.

COR
is the correctable error count (in decimal).

UNC
is the uncorrectable error count (in decimal).

```
User:   ZSONS D E C 47 5

System: SONS0004I 09.40.38
              MOD TYPE     SDA  USE DUP STAT  VSN   LOCK COR   UNC
              047 3380     0EE5 RLT 048   ON BP0001 NONE 00000 00003
              048 3380     0463 RLT 047   ON BP0002 NONE 00000 00003
        DISPLAY COMPLETE
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS DISPLAY ERROR LIMITS–Display Error Limits

Use this command to display the current error limit count and time interval.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Display Error Limits──────────────────────────────────────►◄
```

Additional Information

Enter the ZSONS ALTER ERROR LIMITS command to change the current error limit count and time interval.

Examples

The current error limit count and time interval are displayed in the following example.

```
User:   ZSONS D E L

System: SONS0003I 09.40.59 ERROR LIMITS COUNT-   0, INTERVAL-   0
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS DISPLAY HALT–Display Halt Timeout Value

Use this command to display the time interval that the DASD missing interrupt handler (DMIH) waits before stopping and restarting requests on the module queues after a 5-second timeout interval ends.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Display Halt──────────────────────────────────────────────►◄
```

Additional Information

Enter the ZSONS ALTER HALT command to change the timeout value for the lost-interrupt recover routine in the DMIH.

Examples

The lost interrupt timeout value is displayed in the following example.

```
User:   ZSONS D H

System: SONS0001I 09.40.59 HALT TIMEOUT VALUE IS  15
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS DISPLAY SCAN–Display Scan Time Value

Use this command to display the time value that specifies how often the DASD module queues are scanned for I/O requests that are not being processed by the TPF system in a timely method.

Requirements and Restrictions

None.

Format

```
►►──ZSONS Display Scan──────────────────────────────────────────────►◄
```

Additional Information

- The scan time interval is originally set to 0 by the TPF system, which indicates that the module queues are actually scanned every 5 seconds.
- Enter the ZSONS ALTER SCAN command to change the time interval for scanning the module queues.

Examples

The time value for the module scan is displayed in the following example.

```
User:   ZSONS D S

System: SONS0002I 09.40.59 SCAN TIME VALUE IS  15
```

Related Information

See *TPF Database Reference* for more information about DASD support.

ZSONS DISPLAY SCP–Display RCS State Change Pending Timeout Value

Use this command to display the time interval that the record cache subsystem (RCS) state change pending monitor (SCPM) waits before initiating action to take the device offline.

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format

```
►►──ZSONS Display SCP──────────────────────────────────────────────►◄
```

Additional Information

Enter the ZSONS ALTER SCP command to change the timeout value for the RCS SCPM routine.

Examples

The state change pending timeout value is displayed in the following example.

```
User:   ZSONS D SCP

System: SONS0007I 13:22:31 RCS STATE-CHANGE-PENDING TIMEOUT VALUE IS 15
```

Related Information

See *TPF Database Reference* for more information about DASD support.

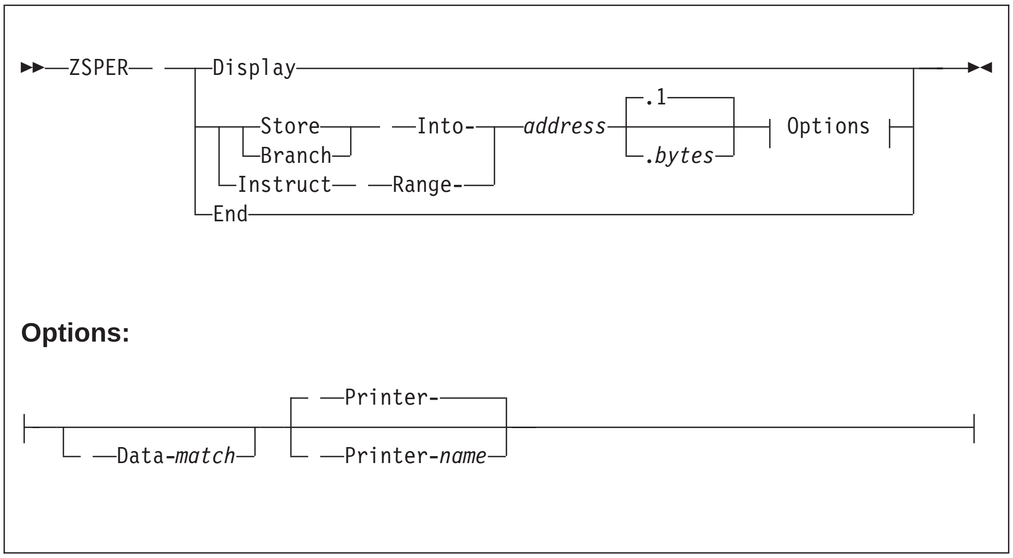
ZSPER–Alter and Display PER Options

Use this command to change or display the program event recording (PER) options online. These options are used to display status, and activate and deactivate the hardware PER facility of an Enterprise Systems Architecture (ESA) processor.

Requirements and Restrictions

- Only one of the supported PER functions can be active at a time.
- You should be extremely careful when using the PER facility on a production system because it can seriously affect system performance.

Format



Display
displays information that indicates whether or not the PER facility is active.

Store Into
traces PER storage alteration events.

Branch Into
traces successful PER branching events. This parameter is valid only for ESA/390 processors.

Instruct Range
traces PER instruction fetching events.

address
is the 1- to 8-digit hexadecimal storage address to be monitored.

bytes
is the 1- to 8-digit hexadecimal number of bytes to be monitored. Address wrap-around is not supported.

Data-*match*
processes the PER interrupt only if the instruction at the PER interrupt address matches the specified data, where *match* is the 2- to 16-digit hexadecimal data. Intervening blanks are not allowed.

Note: The PER interrupt count is incremented regardless of the result of the comparison.

Printer-*name*
sends the output to the specified device, where *name* is the 1- to 8-character alphanumeric device name. Intervening blanks are not allowed. The default device is the read-only (RO) computer room agent set (CRAS).

Note: The value specified for this parameter is passed with the PER interrupt data to the UPER real-time user exit. You must create the program necessary to support the specified device.

End
stops the PER facility.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZSPER HELP
  ZSPER ?
- A PER interrupt provides the instruction address used to fetch the current instruction when a PER event is recognized. When the instruction is the target of EXECUTE, the instruction address used to fetch the EXECUTE instruction is provided.
- In a test environment under VM, use the CP TRACE facility. The PER facility that is provided by the TPF system is a subset of the ESA PER capability and PER interrupts do not stop the operation of the TPF system.
- ESA/370 processors provide PER storage alteration and instruction fetching functions in the TPF system.
- ESA/390 processors provide PER storage alteration, instruction fetching, and branching functions support in the TPF system.
- z/Architecture support processors provide PER storage alteration, instruction fetching, and branching functions support in the TPF system.
- The PER facility is automatically stopped if more than 10 PER interrupts are received in any 10-second interval.

Examples

The following example monitors PER branches into addresses X'1A480'–X'1A485'.

```
User:   ZSPER BRANCH INTO-1A480.6

System: SPER0002I 09.20.20 PER TRACE STARTED
```

The following example stops the PER facility.

```
User:   ZSPER END

System: SPER0003I 09.30.06 PER TRACE ENDED
```

Related Information

See *TPF Program Development Support Reference* or *ESA/390 Principles of Operation* for more information about the program event recording (PER) facility.

ZSQLD–Display and Maintain the Structured Query Language Database Management System Directory

Use this command to maintain and display the structured query language (SQL) database management system (DBMS) directory. You can perform the following functions:

- Define a new relational database
- Change an existing relational database definition
- Delete an existing relational database definition
- Display information about a relational database
- Initialize the SQL DBMS directory.

Requirements and Restrictions

The TPF Application Requester (TPFAR) feature must be installed on the TPF system.

Format

```
►►──ZSQLD──  ──────────────────────────────────────────────────────────────►

                                           (1)
►──┬──┬─Add────┬── ──Rdb-rdbname── ──┬──────Lu-luname──┬──────────────────┬─┬──►◄
   │  └─Modify─┘                     │                 └─┤ LU Options ├───┘ │
   │                                 │     (1)                             │
   │                                 └──────Hostname-hostn──┬──────────────┬┘
   │                                                        └─┤ IP Options ├┘
   ├─Display── ──┬─Long──┬── ──Rdb-rdbname──────────────────────────────────┤
   │             └─Short─┘                                                 │
   ├─INITialize────────────────────────────────────────────────────────────┤
   ├─Remove── ──Rdb-rdbname────────────────────────────────────────────────┤
   └─Trace── ──┬─ALL─┬─────────────────────────────────────────────────────┘
               └─SEL─┘
```

LU Options:

```
                    (2)                                    (2)
   ┌─ ──MOde-TPFDB2MO──────┐               ┌─ ──CCsid-500─────────────────┐
├──┴──────────────────────┴──┬───────────────┬──┴──────────────────────────────┴──►
   └─ ──MOde-mode──────────┘ └─ ──Netid-netid─┘ └─ ──CCsid-──┬─ccsid──────────────┤
                                                        └─ccsid·ccsid·ccsid─┘

                 (2)                          (2)              (2)
  ┌─ ──TPFCCsi-500──────────────────┐ ┌─ ──SELect-ON──┐ ┌─ ──MAxhc-0──────┐
►─┴─────────────────────────────────┴─┴───────────────┴─┴─────────────────┴──┤
  └─ ──TPFCCsi-──┬─ccsid──────────────┤ └─ ──SELect-──┬─ON──┤ └─ ──MAxhc-maxhc──┘
              └─ccsid·ccsid·ccsid─┘              └─OFF─┘
```

IP Options:

```
                (2)                                    (2)
   ┌─ ──CCsid-500──────────────────┐ ┌─ ──TPFCCsi-500─────────────────┐
├──┴───────────────────────────────┴─┴─────────────────────────────────┴──────►
   └─ ──CCsid-──┬─ccsid──────────────┤ └─ ──TPFCCsi-──┬─ccsid─────────────┤
             └─ccsid·ccsid·ccsid─┘                └─ccsid·ccsid·ccsid─┘

               (2)                 (2)                  (2)
  ┌─ ──SELect-ON──┐ ┌─ ──MAxhc-0──────┐ ┌─ ──POrt-446────┐
►─┴───────────────┴─┴─────────────────┴─┴────────────────┴──────────────────┤
  └─ ──SELect-──┬─ON──┤ └─ ──MAxhc-maxhc──┘ └─ ──POrt-port───┘
             └─OFF─┘
```

Notes:

1 This parameter is optional when you specify the MODIFY parameter.

2 The default values apply only when you specify the ADD parameter.

Add
defines a new relational database.

Modify
changes an existing relational database definition.

Attention: This parameter can deallocate all conversations in the hot conversation or hot connection (hotcon) table (HCT) or close all sockets in the HCT.

Rdb-*rdbname*
is the 1- to 18-character alphanumeric name of a remote relational database.

Note: If you specify the DISPLAY SHORT or REMOVE parameters, you can use an asterisk (*) as a wildcard character for this parameter.

Lu-*luname*
is the 1- to 8-character alphanumeric name of the logical unit where the relational database is located.

Note: The logical unit must be a primary remote LU 6.2 resource.

Hostname-*hostn*
is the 1- to 128-character alphanumeric host name of the server or a dotted decimal notation of the Internet Protocol (IP) address of the server.

MOde-*mode*
is 1- to 8-character alphanumeric name of the Advanced Program-to-Program Communications (APPC) mode used with the relational database when an APPC allocate verb is issued.

Netid-*netid*
is the 1- to 8-character alphanumeric ID of the network where the relational database is located.

Note: To clear the network ID, you must specify the MODIFY parameter and specify an asterisk (*) for the NETID parameter.

CCsid-*ccsid*
is the coded character set identifier (CCSID) for the relational database in the range 0–65 536.

The CCSID for the server is what you specify until the server completes connection. After the server completes connection, the CCSID is the coded character set (or sets) actually used by the server.

CCSIDs are specified by a single CCSID or three CCSIDs connected by periods. If you specify a single CCSID, the corresponding relational database must be a single-byte or a multi-byte system.

CCsid-*ccsid·ccsid·ccsid*
are the coded character set identifiers (CCSIDs) for the relational database in the range 0–65 536.

The CCSID for the server is what you specified until the server completes connection. After the server completes connection, the CCSID is the coded character set (or sets) actually used by the server.

CCSIDs are specified either by a single CCSID or three CCSIDs connected by periods (*ccsid*.*ccsid*.*ccsid*). If you specify several CCSIDs, the corresponding relational database is assumed to be a mixed-byte system. If you specify a mixed-byte system, the CCSIDs specified must be a legal combination of single-byte, double-byte, and mixed-byte CCSIDs in that order.

TPFCCsi-*ccsid*
is the coded character set identifier (CCSID) for the TPF system database in the range 0–65 536. The same considerations described for CCSIDs also apply to the TPFCCsi parameter.

TPFCCsi-*ccsid·ccsid·ccsid*
is the coded character set identifier (CCSID) for the TPF system database in the range 0–65 536. The same considerations described for CCSIDs also apply to the TPFCCsi parameter.

MAxhc-*maxhc*
is the maximum number of hot conversations or hot connections (hotcons) allocated to the relational database in the range 0–65 535.

Display
displays information about a relational database.

Long
displays summary information and diagnostic information about a relational database.

Short
displays only summary information about one or more relational databases.

INITialize
initializes the SQL DBMS directory.

Remove
deletes one or more relational database definitions.

Trace
controls the retention of SQL trace entries. Trace information from the SQL communications area is retained in a circular buffer. You can display this information by using the ZSTTD command. Specify one of the following:

ALL
specifies that global tracing is active, so that all request and response traffic with a remote database is retained. If tracing is not set, it defaults to ALL.

SEL
specifies that selective tracing is active and only request/response traffic for a specified relational database is retained. Databases are specified by using the SELECT parameter with the ADD or MODIFY parameters ZSQLD command.

SELect
controls the retention of SQL trace entries for an individual relational database when selective tracing is enabled, where:

ON
means trace entries are retained.

OFF
means trace entries are not retained.

The default is ON. If selective tracing is enabled and no relational databases have been selected, no trace entries are retained.

For example, if ZSQLD TRACE SEL is issued, followed by ZSQLD MODIFY RDB-DB23PRD SEL-ON, selective tracing is enabled for DB23PRD relational database traffic. Specifying ZSQLD MODIFY RDB-DB23PRD SEL-OFF stops retention of the traffic between the TPF system and the remote server.

POrt-*port*
is the port number for the database on the remote application server in the range 0–65 536.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZSQLD HELP
  ZSQLD ?
- To switch from Advanced Program-to-Program Communications (APPC) to Transmission Control Protocol/Internet Protocol (TCP/IP), enter the ZSQLD command with the MODIFY and HOSTNAME parameters specified; the values for LU name, net ID, and mode that were previously associated with APPC will no longer be used.
- To switch from TCP/IP to APPC, enter the ZSQLD command with the MODIFY and LUNAME parameters specified; the host name previously associated with TCP/IP will no longer be used.
- The MAXHC parameter refers to *hotcons*, which represent conversations in APPC and connections in TCP/IP. The socket sweeper will not clean up the socket associated with a hotcon in TCP/IP.
- When connected to more than one physically separate network, ensure that the specified host name resolves to an IP address that points to the correct network. You can use the TPF IP routing table for this condition.

Examples

A new relational database with a specified host name is defined in the following example.

```
User:   ZSQLD A R-DB23TST HOST-9.117.147.166

System: SQLD0011I 17:09:22 SDD ENTRY FOR RDB-DB23TST ADDED
```

The SQL DBMS directory is initialized in the following example.

```
User:   ZSQLD INITIALIZE

System: SQLD0014I 10:38:33 SDD INITIALIZED
```

A new relational database is defined in the following example.

```
User:   ZSQLD A R-DB2A LU-LU62AAA MAXHC-12

System: SQLD0011I 10:40:12 SDD ENTRY FOR RDB-DB2A ADDED
```

In the following example, a relational database with a mixed-byte CCSID is defined with the TPF system also having a mixed-byte CCSID and with selective tracing being enabled for it.

```
User:   ZSQLD A R-DB2A LU-LU62AAA CC-1027.4396.5035 TPFCC-1027.4396.5035 SEL-ON

System: SQLD0011I 10:40:12 SDD ENTRY FOR RDB-DB2A ADDED
```

Summary information about all the relational databases is displayed in the following example, where:

RDB
: is the name of the remote relational database.

NETID
: is the network ID of the remote relational database, or N/A if this relational database is accessed by TCP/IP.

LU
: is the logical unit name of the remote relational database, or N/A if this relational database is accessed by TCP/IP.

MAXHC
: is the maximum number of hot conversations or hot connections allocated to the remote relational database.

SEL
: is the selective trace status (either ON or OFF).

HOSTNAME
: is the name of the host or IP address of the server, or N/A if this relational database is accessed by TCP/IP.

```
User:   ZSQLD DIS S RDB-*

System: SQLD0118I 10.48.54
        START OF ZSQLD DISPLAY
        RDB                NETID    LU        MAXHC  SEL  HOSTNAME
        ---                -----    --        -----  ---  --------
        TPFDB              N/A      N/A       3      ON   9.117.147.166
        DB23TST                     DB2TESTP  0      ON   N/A
        DB23PRD            TPFNOD1  DB2PRDS   2      OFF  N/A
        END OF ZSQLD DISPLAY
```

Summary and diagnostic information about the DB2TST relational database is displayed in the following example, where:

RDB
: is the name of the remote relational database.

NETID
: is the network ID of the remote relational database, or N/A if this relational database is accessed by TCP/IP.

LU
: is the logical unit name of the remote relational database, or N/A if this relational database is accessed by TCP/IP.

HOSTNAME
: is the name of the host or IP address of the server, or N/A if this relational database is accessed by TCP/IP.

CCSID
: is the coded character set ID.

TPFCCSID
: specifies the TPF system coded character set identifiers.

SEL
: is the selective trace status (either ON or OFF).

| **POrt-***port*
| : is the port number for the database on the remote application server.

MAXHC
: is the maximum number of hot conversations or hot connections (hotcons) allocated to the remote relational database.

SRVRLSLV
: is the server release level.

SRVRNAM
: is the server name.

SRVCLSNM
: is the server class name.

TRACE
: is the trace status.

MODE
: is the APPC mode name, or N/A if this relational database is accessed by TCP/IP.

AVAIL HC
: is the number of available hot conversations or hot connections.

NEXT HC ADDR
: is the next hot conversation or hot connection entry address.

```
User:   ZSQLD DIS L RDB-TPFDB

System: SQLD0117I 10.48.54
        START OF ZSQLD DISPLAY
        RDB                 NETID     LU       HOSTNAME
        ---                 -----     --       --------
        TPFDB               N/A       N/A      9.117.147.166

        CCSID (S.D.M.)      TPFCCSID (S.D.M.)  MAXHC  SEL   PORT
        -----               --------           -----  ---   ----
         1252.    0.    0     500.    0.    0      3   ON    446
        SRVRLSLV - DB2 UDB
        SRVNAME  - DB2
        SRVCLSNM - QDB2/NT
        TRACE    - ALL
        MODE     - N/A
        AVAIL HC -     1     NEXT HC ADDR - 056CD010
        END OF ZSQLD DISPLAY
```

The specified relational database is deleted in the following example.

```
User:   ZSQLD R R-DB2

System: SQLD0016I 10:52:52 SDD ENTRIES MATCHING RDB-DB2 REMOVED
```

The logical unit and network ID for the specified relational database is changed in the following example.

```
User:   ZSQLD M R-RDB LU-TPFDB2T NETID-01234567

System: CSMP0097I 09.55.30 CPU-B SS-BSS  SSU-HPN  IS-01
        SQLD0015I 09.55.30 SDD ENTRY FOR RDB-RDB MODIFIED
```

Related Information

- See the *TPF Application Programming* and the *Character Data Representation Architecture Reference and Registry* for more information about the database management system (DBMS), relational databases, and coded character set identifiers.
- See the *TPF Transmission Control Protocol/Internet Protocol*for more information about TCP/IP.

ZSSLD–Manage Secure Sockets Layer (SSL) Daemon Processes

The information for this command is delivered as browser-readable HTML files only. To view this information, go to http://www.ibm.com/tpf/pubs/tpfpubs.htm, click **SSL for the TPF 4.1 System: An Online User's Guide**, and click **Using Commands** from the left navigation bar.

ZSTAT–Display System Status

Use this command to do the following:

- Display status information about the TPF system, such as the number of allocated and available main storage blocks and the number of active entry control blocks (ECBs).
- Generate time-initiated displays of the status information.
- Display information about I-stream utilization and work list counts.
- Display information about block usage by owner.

Requirements and Restrictions

None.

Format

```
►►──ZSTAT──┬─────────────┬──────────────────────────────────────►◄
           ├──Dinterval──┤
           ├──STOP───────┤
           └──U──────────┘
```

D*interval*
: generates a time-initiated display, where *interval* is the number of minutes, from 1–99, between displays. An immediate response is sent to the prime CRAS. The time-initiated responses are sent only to the read-only (RO) console.

STOP
: stops a time-initiated display.

U displays information about I-stream utilization and work list counts.

Additional Information

- Enter ZSTAT (without any parameters) to display status information about the TPF system.
- If you do an initial program load (IPL) while a time-initiated display is active, the display is resumed with the specified time interval when the IPL is completed.
- To generate a time-initiated display for the I-stream utilization information, use the ZSTIM command.

Examples

The following example displays status information about the TPF system. Only one subsystem is active.

Note: The TCP/IP input and output counts are weighted message counts. See *TPF Transmission Control Protocol/Internet Protocol* for more information about message counts for TCP/IP applications.

```
User:   ZSTAT

System: STAT0012I 14.04.03 SYSTEM STATUS DISPLAY
                      IOB  FRAME COMMON    SWB    ECB
        ALLOCATED    2704  20000    276   1252    500
        AVAILABLE    2704  18858    272   1214    495

        SYSTEM HEAP FRAMES      1021
        THREAD FRAME PENDING       0

        ACTIVE ECBS                5
        DLY/DFR ECBS               0
        PROCESSED                336
        LOW SPEED                  0
        ROUTED                     0
        CREATED                35469
        SNA                        0
        TCP/IP INPUT            3409
        TCP/IP OUTPUT           2590
        END OF DISPLAY
```

The following example displays information about I-stream utilization and work list counts.

```
User:   ZSTAT U

System: STAT0011I 13.39.46 SYSTEM UTILIZATION DISPLAY
        NUM  ADDR  UTIL/ ADJ   CROSS READY INPUT   VCT SUSPD DEFER ACT-ECB
        IS-1  0   100.0/ 93.7     1     2    18     3     0    10     34
        IS-2  1    99.7/ 93.8     0     0    11     1     2     9     24
        IS-3  2    99.9/ 93.8     7     0     3     2     0     8     20
```

Related Information

See *TPF Concepts and Structures* for general information about the TPF system.

ZSTIM A–Add a Time- or Stage-Initiated Message

Use this command to add messages to the time-initiated message table. You can add a command, a message to start an application, a text message to send to a functional support console (FSC), or any other meaningful message.

Messages can be time-initiated or stage-initiated. *Time-initiated* messages are issued at a specific time during the day. *Stage-initiated* messages are issued when the TPF system reaches a specific stage; for example, at the end of restart.

Requirements and Restrictions

You can add a time-initiated message to the table only when the TPF system is above 1052 state because clocks must be running. You can add stage-initiated messages at any time.

Format

```
►►──ZSTIM A── ──FReq=──┬─AT─────┬──,TIme=──┬─time───┬──,MSg=──┬─────┬─text─┬─────┬──────────►
                       ├─IN─────┤          ├─1052S──┤         └─'───┘      └─'───┘
                       ├─EVERY──┤          ├─1052E──┤
                       └─DAILY──┘          ├─CYCLU──┤
                                           ├─CYCLD──┤
                                           └─NORM───┘

                                   ┌─,PAST=Y─┐
►──┬─────────────────────────┬─────┼─────────┼───────────────────────────────────────────►◄
   ├─,FSC=──┬─number──┬──────┤     └─,PAST=N─┘
   │        ├─AUDT────┤      │
   │        ├─COMM────┤      │
   │        ├─DASD────┤      │
   │        ├─PRC─────┤      │
   │        ├─RDBS────┤      │
   │        ├─RO──────┤      │
   │        └─TAPE────┘      │
   └─,APL=name───────────────┘
```

FReq
: is one of the following options that indicates the frequency with which the message is issued.

AT
: issues the message at the specified time.

IN
: issues the message in the specified number of minutes.

EVERY
: issues the message at a repeating time interval, which is specified in minutes.

DAILY
: issues the message each day at the specified time.

TIme
: specifies the time or time interval to issue the message.

time
is the time or time interval. For the AT or DAILY option, specify the time in the *hhmm* format, where *hh* is hours and *mm* is minutes. For the IN or EVERY option, specify the time interval in the *mm* format.

1052S
issues the message at the beginning of restart. You can specify this option only for the AT or EVERY option.

1052E
issues the message at the end of restart. You can specify this option only for the AT or EVERY option.

CYCLU
issues the message after the TPF system is cycled from a lower state to a higher state; for example, from 1052 state to MESW state. You can specify this option only for the AT or EVERY option.

CYCLD
issues the message after the TPF system is cycled from a higher state to a lower state; for example, from NORM state to 1052 state. You can specify this option only for the AT or EVERY option.

NORM
issues the message when the TPF system reaches NORM state. You can specify this option only for the AT or EVERY option.

MSg=*text*
is the message to be issued, where *text* is 1–72 characters.

If you enclose the text in single quotations ('), the text can contain all characters except single quotations. If you do not enclose the text in single quotations, the text cannot contain commas (,) and plus signs (+).

Attention: The message that you specify is not verified. Therefore, the results cannot be predicted if the message text is not correct.

FSC
sends the message to one of the following functional support consoles (FSCs):

number
is a decimal number from 1–16 that represents the FSC.

AUDT
sends the message to the AUDT FSC.

COMM
sends the message to the COMM FSC.

DASD
sends the message to the DASD FSC.

PRC
sends the message to the PRC FSC.

RDBS
sends the message to the real-time database services (RDBS) FSC, which is used by the TPF Application Requester (TPFAR) feature.

RO
sends the message to the RO FSC.

TAPE
sends the message to the TAPE FSC.

APL=*name*
sends the message to the specified application, where *name* is the 4-character alphanumeric name of the application. The first character must be alphabetic.

Note: You must ensure that the application name is valid and that the specified application can process the message at the specified time.

PAST
specifies whether or not to process time-initiated messages that are past due. If you specify PAST=N, the message that is past due is discarded if it cannot be processed within 1 minute after it is issued.

A *past-due* time-initiated message is a message that was not issued at the specified time. For example, if the processor was offline when the time-initiated message was to be issued, that message is past-due.

Additional Information

- The order of issuing the time-initiated messages may change depending on the time it takes to enter them into the time-initiated message table, the activation time, and the processing time. Therefore, the messages may not be issued in the intended sequence. Stage-initiated messages are issued in the order that they are entered in the table.
- This function does not verify the message content in any way. Therefore, it is your responsibility to ensure that the resources required by the message are available and that the syntax of the message is correct.
- If you specify the AT option or the IN option for the FREQ parameter, the message is deleted from the time-initiated message table after it is issued, or if it is past-due and you do not process it (PAST=N). If you specify the EVERY option or the DAILY option, the message remains in the time-initiated message table until you delete it using the ZSTIM C command.
- Message routing is determined by the following criteria:
  – If the routing is specified by FSC, it is not necessary to do any further destination determination.
  – If the message starts with a ’Z’ it is assumed to be a command:
    - If the routing is specified by APL=SMPx, the message is edited for basic format.
    - If the routing is not specified, the message is edited for basic format and sent to SMP in this processor.
    - If the routing is specified but is not SMPx, the message is forwarded to the application. It is not edited for basic format.
  – If the message does not start with a ’Z’ it may be a prefixed command:
    - If the routing is specified by APL=SMPx, the message is assumed to be prefixed and is edited for basic format.
  – Otherwise
    - If the routing is specified by APL, it is forwarded to the indicated application.
    - If the routing is not specified, it is forwarded to the RES0 application.

Examples

In the following example, a message is added to the time-initiated message table to display system status every 5 minutes.

```
User:   ZSTIM A FREQ=EVERY,TIME=05,MSG=ZSTAT

System: STMA0001I 08.39.13 MSG ADDED TO TIM TBL
```

In the following example, a message is added to the time-initiated message table to display tape status every day at 3 p.m.

```
User:   ZSTIM A FREQ=DAILY,TIME=1500,MSG=ZDTAP

System: STMA0001I 08.39.13 MSG ADDED TO TIM TBL
```

In the following example, a message is added to the time-initiated message table to change the system error options at the end of each restart.

```
User:   ZSTIM A FREQ=EVERY,TIME=1052E,MSG='ZASER HALT,DUPL,ALL BLKS'

System: STMA0001I 08.39.13 MSG ADDED TO TIM TBL
```

In the following example, a message is added to the time-initiated message table to display system status in 30 minutes. The ZSTAT command is discarded if it cannot be processed within 1 minute after it is issued.

```
User:   ZSTIM A FREQ=IN,TIME=30,PAST=N,MSG=ZSTAT

System: STMA0001I 08.39.13 MSG ADDED TO TIM TBL
```

In the following example, a message is added to the time-initiated message table to display a file record on a specific subsystem user.

```
User:   ZSTIM A FREQ=IN,TIME=1,APL=SMPB, MSG='SSU/ZDFIL 07F40001

System: STMA0001I 08.39.13 MSG ADDED TO TIM TBL
```

Related Information

None.

ZSTIM C–Cancel a Time- or Stage-Initiated Message

Use this command to remove a message from the time-initiated message table. You can cancel a permanent message or a message that has not yet been issued.

Requirements and Restrictions

None.

Format

```
►►──ZSTIM C── ──index──────────────────────────────────────────────►◄
```

index
is the 1- to 3-digit decimal index number associated with the message you want to cancel.

Additional Information

- Enter the ZSTIM D command to display the messages in the time-initiated message table and the associated index numbers.
- Index numbers are reassigned each time a message is removed from the time-initiated message table. Therefore, enter the ZSTIM D command before you remove messages from the table to ensure that you specify the correct index number.

Examples

The message with index number 003 is canceled in the following example. Notice that the index numbers are reassigned to the messages after message 003 is canceled.

```
User:   ZSTIM D

System: STMD0001I 13.51.52 TIME-INITIATED MSG TBL
        001 14:01   SMPB    ZRPGM CAPH L
        002 14:47   CVZZ    ZRPGM CAPI L
        003 14:51   TAPE    ZDPGM CAPR
        004 18:00*  SMPB    ZRPGM CAPR L
        005 20:00*  SMPB    ZRPGM CAPP L
        006 23:00*N SMPB    ZSTAT U
        007 06:00*X SMPB    ZSTAT
        END OF TIM TBL DISPLAY

User:   ZSTIM C 003

System: STMC0001I 08.44.48 MSG CANCELLED FROM TIM TBL

User:   ZSTIM D

System: STMD0001I 13.51.52 TIME-INITIATED MSG TBL
        001 14:01   SMPB    ZRPGM CAPH L
        002 14:47   CVZZ    ZRPGM CAPI L
        003 18:00*  SMPB    ZRPGM CAPR L
        004 20:00*  SMPB    ZRPGM CAPP L
        005 23:00*N SMPB    ZSTAT U
        006 06:00*X SMPB    ZSTAT
        END OF TIM TBL DISPLAY
```

Related Information

None.

ZSTIM D–Display Time-Initiated Message Table

Use this command to display the messages in the time-initiated message table and their associated index number.

Requirements and Restrictions

If the time-initiated message table is updated, you must display the entire table before you can display a specific message from the message table.

Format

```
►►──ZSTIM D──┬─────────┬───────────────────────────────────────────►◄
             └─index───┘
```

index
: is the 1- to 3-digit decimal index number associated with the message you want to display. If you do not specify an index number, the entire time-initiated message table is displayed.

Additional Information

- Index numbers are reassigned each time a message is removed from the time-initiated message table.
- Only the first 30 characters of each message are displayed when you display the entire time-initiated message table.
- Enter the ZSTIM C command to remove a message from the time-initiated message table.
- Enter the ZSTIM A command to add a message to the time-initiated message table.
- If the system clocks are not running, **—:—** is displayed in place of the message times for time-initiated messages.

Examples

The following information is displayed for each line in the example:

1 to 3
: is the index number of the message.

5 to 9
: is the time the message will be issued.

10 indicates the frequency that the message is issued.

blank
: The message is issued once at the specified time.

*
: The message is issued at a repeating time interval, or the message is issued daily at the specified time.

11 is one of the following:

blank
: Indicates that the message will be processed if it becomes past due.

N Indicates that the message will ***not*** be processed if it becomes past due.

X Indicates that the message will be issued the next day at the specified time.

13 to 16
is the destination of the message.

18 to 19
is the repeating time interval, in minutes, for issuing the message.

21 to 50
is the message text.

The following example displays the entire message table.

```
User:   ZSTIM D

System: STMD0001I 13.51.52 TIME-INITIATED MSG TBL
        001 14:01   SMPB    ZRPGM CAPH L
        002 14:30*  SMPB 60 ZDTAP
        003 14:47   CVZZ    ZRPGM CAPI L
        004 14:51   TAPE    ZDPGM CAPR
        005 18:00*  SMPB    ZRPGM CAPR L
        006 20:00*  SMPB    ZRPGM CAPP L
        007 23:00*N SMPB    ZSTAT U
        008 06:00*X SMPB    ZSTAT
        END OF TIM TBL DISPLAY
```

Related Information

None.

ZSTIM I–Initialize Time-Initiated Message Table

Use this command to initialize the time-initiated message table.

Requirements and Restrictions

None.

Format

```
►►──ZSTIM I──────────────────────────────────────────────────────────►◄
```

Additional Information

This command is a useful debugging or recovery tool for the system alerts time-initiated message function.

Examples

The time-initiated message table is initialized in the following example.

```
User:   ZSTIM I

System: STMI0001I 13.39.46 TIM TBL INITIALIZED
```

Related Information

None.

ZSTOP–Stop Macro Tracing or Macro Counting for the Real-Time Trace Utility

Use this command to stop the macro tracing or macro counting function of the real-time trace (RTT) utility. The RTT utility counts or traces the use of specific macros and related system activity.

Use the ZCNTM command to start the macro counting function. Use the ZTRAC command to start the macro tracing function.

Requirements and Restrictions

Each ZCNTM or ZTRAC command must have a corresponding ZSTOP command. If you enter a second ZCNTM or ZTRAC command before the count or trace period is ended with a ZSTOP command, the second ZCNTM or ZTRAC command is rejected.

Format

```
►►──ZSTOP──────────────────────────────────────────────────────────────►◄
```

Additional Information

All output from the RTT utility is written to the real-time tape (RTL/RTA). Use the offline diagnostic output formatter (DOF) utility to process the RTT output.

Examples

In the following example tracing is stopped.

```
User:   ZSTOP

System: RTT STOPPED BY SS: BSS
```

Related Information

- See “ZCNTM–Start Macro Counting for the Real-Time Trace Utility” on page 224 for more information about starting the macro counting function.
- See “ZTRAC–Start Macro Tracing for the Real-Time Trace Utility” on page 1376 for more information about starting the macro tracing function.
- See *TPF Program Development Support Reference* for more information about the RTT utility and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline DOF utility.

ZSTRC–Alter and Display System Trace Options

Use this command to change or display the system trace options.

Requirements and Restrictions

You can only enter this command from the basic subsystem (BSS).

Format

```
                          ┌──────────────────────────┐
                          │ (1)                      │
                          ▼                          │
►►──ZSTRC── ──┬─Alter── ──────────┬────┬─┬─BLKCHK───┬─┴──┬──────────────►◄
              │                   └─NO─┘ ├─BRANCH───┤    │
              │                          ├─CDEBUG───┤    │
              │                          ├─ENTER────┤    │
              │                          ├─IDLETMR──┤    │
              │                          ├─IO───────┤    │
              │                          ├─MACRO────┤    │
              │                          ├─REGS─────┤    │
              │                          ├─STACK────┤    │
              │                          ├─STATIC───┤    │
              │                          ├─SYSLOG───┤    │
              │                          ├─VRLOG────┤    │
              │                          └─XHOOKS───┘    │
              └─Display──────────────────────────────────┘
```

Notes:

1 You can also separate the parameters with a comma (,).

Alter
changes the system trace options.

NO
turns off the system trace options.

BLKCHK
starts block check mode. Block check mode is used to flag certain coding errors, such as writing beyond the end of a block, passing blocks chained to other blocks, and using storage that has already been released. Performance is decreased while running block check mode.

BRANCH
starts tracing the BALR, BASR, BAKR, and BASSM branch instructions. The target of the branch instructions is displayed in the branch trace table, not the address of the branch instruction.

CDEBUG
enables the C function trace for ISO-C programs compiled using the TEST option of one of the IBM C/370 family of compilers supported by the TPF 4.1 system.

NOCDEBUG disables the C function trace for ISO-C programs compiled using the TEST option.

ENTER
starts the enter/back trace, which includes ENTxC and BACKC program statements in the macro trace.

Note: When you start the enter/back trace, the macro trace also starts.

IDLETMR
turns on the idle timer. When the TPF systems are idle and running under the IBM Virtual Machine (VM) system or shared PR/SM, the idle timer option causes idle TPF systems to use fewer central processing unit (CPU) cycles than they normally would while idle. By reducing the amount of CPU cycles consumed by idle TPF systems, the IBM VM system and shared PR/SM can support additional TPF systems or better service other work being performed by the IBM VM system or shared PR/SM.

The idle timer option causes TPF systems to use fewer CPU cycles by decreasing the frequency of CPU timer interrupts on idle I-streams. It decreases the frequency of CPU timer interupts for a given I-stream only when that I-stream is idle. When an I-stream is performing work or when the idle timer option is turned off, CPU timer interrupts occur at their regular frequency.

When the TPF system is running native or under dedicated PR/SM, the idle timer option is ignored and CPU timer interrupts occur at their regular frequency.

IO starts the I/O trace, which traces I/O interrupts and CIO macros associated with an I/O device.

MACRO
starts the macro trace, which traces all the SVC macros associated with an ECB. Fast-linked macros are not traced.

Note: When you stop the macro trace (NOMACRO), the enter/back trace also stops.

REGS
includes registers in the macro trace. Therefore, you must also start the macro trace (MACRO).

Note: This change does not take effect until the next initial program load (IPL) is performed. Enter **ZSTRC DISPLAY** to display the value of the REGS parameter for this IPL and for the next IPL.

STACK
places up to 68 bytes of the C function stack data in the trace table when the CDEBUG parameter is active. The information is displayed in the dump.

If you specify NOSTACK, only the address of the stack area is placed in the trace table when the CDEBUG parameter is active.

If the CDEBUG parameter is not active, STACK or NOSTACK will not trace any information.

STATIC
places up to 68 bytes of the C function static data in the trace table when the CDEBUG parameter is active. The information is displayed in the dump.

If you specify NOSTATIC, only the address of the static area is placed in the trace table when the CDEBUG parameter is active.

If the CDEBUG parameter is not active, STATIC or NOSTATIC will not trace any information.

SYSLOG
starts tracing the events previously included in macro or I/O traces that are not associated with an ECB or an I/O device. Therefore, you must also start the macro trace (MACRO) or the I/O trace (IO) as appropriate.

VRLOG
activates VEQR mode migration aid logging. This generates a SNAPC dump whenever an ECB-controlled program attempts to access another ECB, or a block owned by another ECB, in VEQR mode.

XHOOKS
enables the C function trace of breakpoints other than *program entry breakpoints* and *program exit breakpoints* for ISO-C programs compiled using the sub-options of the TEST option of one of the IBM C/370 family of compilers supported by the TPF 4.1 system.

NOXHOOKS disables the C function trace of breakpoints other than *program entry breakpoints* and *program exit breakpoints* for ISO-C programs compiled using the sub-options of the TEST option of one of the IBM C/370 family of compilers supported by the TPF 4.1 system.

If the CDEBUG parameter is not active, XHOOKS will not trace any information.

Display
displays the system trace options.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZSTRC HELP
  ZSTRC ?
- The system trace options are initially set as follows:
  – NOBLKCHK
  – BRANCH
  – NOCDEBUG
  – ENTER
  – NOIDLETMR
  – IO
  – MACRO
  – NOREGS
  – NOSTACK
  – NOSTATIC
  – SYSLOG
  – VRLOG
  – NOXHOOK.
- The enter/back trace traces a subset of the macro trace. Therefore, the macro trace is turned on when the enter/back trace is turned on. Also, the enter/back trace is turned off when the macro trace is turned off.
- All system trace options are processor unique. As a result, each option can be turned on or off on each processor without affecting the options specified on other processors. All system trace option settings are preserved across an initial program load (IPL).

Examples

In the following example, the displayed information shows that the macro, enter/back, I/O, branch, and C function trace are turned on, and the block check trace, C function trace STACK trace, C function trace STATIC trace, and C function trace XHOOKS trace, and the idle timer option are turned off.

```
User:   ZSTRC DISPLAY

System: STRC0007I 08.13.39 SYSTEM TRACE OPTIONS
        MACRO TRACE       ON
        ENTER/BACK TRACE  ON
        IO TRACE          ON
        SYSTEM LOG        ON
        BRANCH TRACE      ON
        REGS THIS IPL     OFF
        REGS NEXT IPL     OFF
        BLOCK CHECK       OFF
        VEQR MODE LOGGING ON
        CDEBUG TRACE      ON
        STACK TRACE       OFF
        STATIC TRACE      OFF
        XHOOKS TRACE      OFF
        IDLE TIMER        OFF
```

In the following example, the enter/back, block check, C function trace, C function trace STACK trace, and C function trace XHOOKS trace, and idle timer option are turned on; the I/O and C function trace STATIC trace are turned off.

```
User:   ZSTRC ALTER ENTER NOIO BLKCHK CDEBUG STACK NOSTATIC XHOOKS IDELTMR

System: STRC0007I 15.08.41 SYSTEM TRACE OPTIONS
        MACRO TRACE       ON
        ENTER/BACK TRACE  ON
        IO TRACE          OFF
        SYSTEM LOG        OFF
        BRANCH TRACE      OFF
        REGS THIS IPL     OFF
        REGS NEXT IPL     OFF
        BLOCK CHECK       ON
        VEQR MODE LOGGING OFF
        CDEBUG TRACE      ON
        STACK TRACE       ON
        STATIC TRACE      OFF
        XHOOKS TRACE      ON
        IDLE TIMER        ON
```

Related Information

- See *TPF Program Development Support Reference* for more information about the collated macro trace and I/O trace.
- See *TPF Main Supervisor Reference* for more information about block check mode.
- See *TPF Program Development Support Reference* for more information about the C function trace.

ZSTTD–Display the SQL Trace Table Entries

Use this command to display entries from the structured query language (SQL) trace table on the current subsystem in either of two formats: a summary form or a detail form.

Requirements and Restrictions

The TPF Application Requester (TPFAR) feature must be installed on the TPF system.

Format

```
                          (1)
►►──ZSTTD── ──┬─All─────────┬──┬────────────┬──┬────────────────┬──────────────────►
              └─nbr─────────┘  └─ ──┬─ERR─┬─┘  └─ ──PGM-progname─┘
                                    ├─REP─┤
                                    ├─REQ─┤
                                    └─DET─┘

                        ┌─ ──IS-1───────┐
►─┬──────────────────┬──┼───────────────┼─────────────────────────────────────────►◄
  └─ ──ECB-address───┘  └─ ──IS-istream─┘
```

Notes:

1 The detail (DET) parameter is not used with the ALL parameter.

All
displays all the entries (or the maximum number of 20 entries) in the SQL trace table. This parameter is not used with the DET parameter.

nbr
is the number of entries to display in the SQL trace table, beginning with the entries at the end of the table.

ERR
displays only the replies to SQL commands that were returned with an error; that is, the SQLSTATE value does not equal 0.

REP
displays only the replies to SQL commands.

REQ
displays only the requests for SQL commands.

DET
displays details about the replies to SQL commands that were returned with an error (an SQLSTATE value not equal to 0). This parameter is not used with the ALL parameter.

PGM-*progname*
displays only the SQL trace table entries that were processed by the specified application program, where *progname* is the 4-character alphanumeric name of an application program.

ECB-*address*
displays only the SQL trace table entries that were processed by the ECB at the specified address, where *address* is the 1- to 8-digit hexadecimal system virtual memory (SVM) address.

IS-*istream*
displays entries from the SQL trace table on the specified I-stream, where *istream* is a decimal number from 1 to 16.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:
ZSTTD HELP
ZSTTD ?

Examples

The following information is displayed in the examples:

I O
is the direction of the message, where:

O indicates the SQL request is outbound to the DBMS.

I indicates the SQL request is being returned to the application program.

SQL CODE
is the SQLCODE returned to the application program after the SQL command is processed. The SQLCODE is displayed only for replies to the application program.

SQL STATE
is the SQLSTATE returned to the application program after the SQL command is processed. The SQLSTATE is displayed only for replies to the application program.

SEC NBR
is the section number assigned to the SQL command by the DB2 precompiler.

STMT NBR
is the statement number assigned to the SQL command by the DB2 precompiler.

STMT TYPE
is the statement type of the SQL command.

PGM
is the name of the application program processing the SQL command.

ECB ADDR
is the SVM address of the ECB processing the SQL command.

TIME
is the number of seconds between the time the application program processed the SQL command and the time the application program received the results. The time is displayed only for replies to the application program.

The last 4 entries in the SQL trace table on I-stream 2 are displayed in the following example.

```
User:   ZSTTD 4 IS-2

System: CRDW0004I 09.04.31 SQL TRACE TABLE FOR I-STREAM 2
        I   SQL  SQL   SEC STMT STMT                ECB
        O   CODE STATE NBR  NBR TYPE           PGM  ADDR      TIME
        O               1   58 SELECT         QXOE 2F9A40
        I    100 02000   1   58 SELECT         QXOE 2F9A40   0.054
        O               0   47 COMMIT         QXOE 2F9A40
        I      0 00000   0   47 COMMIT         QXOE 2F9A40   0.175
        END OF TABLE, 4 ENTRIES DISPLAYED
```

The last 3 replies to SQL commands are displayed in the following example.

```
User:   ZSTTD 3 REP IS-2

System: CRDW0004I 08.02.23 SQL TRACE TABLE FOR I-STREAM 2
        I   SQL  SQL   SEC STMT STMT                ECB
        O   CODE STATE NBR  NBR TYPE           PGM  ADDR      TIME
        I      0 00000   0   42 CONNECT TO     QXOE 2F9A40   1.175
        I    100 02000   1   58 SELECT         QXOE 2F9A40   0.154
        I      0 00000   0   47 COMMIT         QXOE 2F9A40   0.075
        END OF TABLE, 3 ENTRIES DISPLAYED
```

The last 3 requests for SQL commands that were processed by the ECB at the specified address are displayed in the following example. Notice that 6 entries were requested but only 3 exist in the SQL trace table.

```
User:   ZSTTD 6 REQ ECB-2F9A40

System: CRDW0004I 11.22.50 SQL TRACE TABLE FOR I-STREAM 1
        I   SQL  SQL   SEC STMT STMT                ECB
        O   CODE STATE NBR  NBR TYPE           PGM  ADDR      TIME
        I               0   42 CONNECT TO     QXOE 2F9A40
        I               1   58 SELECT         QXOE 2F9A40
        I               0   47 COMMIT         QXOE 2F9A40
        END OF TABLE, 3 ENTRIES DISPLAYED
```

All the replies to SQL commands that were processed with an error by the QXOE application program are displayed in the following example. Notice that only one entry exists.

```
User:   ZSTTD ALL ERR PGM-QXOE

System: CRDW0004I 08.02.31 SQL TRACE TABLE FOR I-STREAM 1
        I   SQL  SQL   SEC STMT STMT                ECB
        O   CODE STATE NBR  NBR TYPE           PGM  ADDR      TIME
        I    100 02000   1   58 SELECT         QXOE 2F9A40   0.154
        END OF TABLE, 1 ENTRIES DISPLAYED
```

The DETAIL parameter provides the most recent entries in the trace table. In the following example program QXOE has returned an SQLCODE of 100.

The information returned is defined by the relational distributed architecture. The previous example gives the definitions for the fields, except for SQLCAID, SQLCABC, SQLWARN, SQLERRD, SQLERRP, SQLERRML, and SQLERRMSG, which provide various kinds of information and warnings as defined by the remote database.

```
User:   ZSTTD 1,DET

System: SQL DETAIL TRACE FOR I-STREAM 1
        PROGRAM: QXOE      ECB ADDRESS:   6B1000   RDB: DB23PRD
        SQL_CODE: 100      SQL_STATE: 2000
        SECTION NUM: 00001     STMT NUM: 00058   STMT TYPE: SELECT
        SQLCAID:   SQLCABC: 00000   SQLWARN:
        SQLERRD: 0, 0, 0, 0, 0, 0  SQLERRP:
        SQLERRML: 0  SQLERRMSG:

        END OF TABLE, 1 ENTRIES DISPLAYED
```

The following example shows how the summary trace and the detail trace differ. The fields displayed are defined in a previous example.

```
User:   ZSTTD 8

System:  CRDW0004I 14.40.32 SQL TRACE TABLE FOR I-STREAM 1
         I   SQL  SQL   SEC STMT STMT                  ECB
         O   CODE STATE NBR  NBR TYPE              PGM  ADDR        TIME
         O               0   60 CONNECT TO        QXOE   6D6000
         I -30041 57013   0   60 CONNECT TO        QXOE   6D6000   0.003
         O               1  102 SELECT            QXOE   6D6000
         I  -1024 51007   1  102 SELECT            QXOE   6D6000   0.000
         O               0   79 COMMIT            QXOE   6D6000
         I  -1024 51007   0   79 COMMIT            QXOE   6D6000   0.000
         O               0   79 COMMIT            QXOE   6D6000
         I  -1024 51007   0   79 COMMIT            QXOE   6D6000   0.000
         END OF TABLE, 8 ENTRIES DISPLAYED

User:   ZSTTD 4,DET

System: SQL DETAIL TRACE FOR I-STREAM 1
        PROGRAM: QXOE      ECB ADDRESS:   6D6000    RDB: DB23PRD
        SQL_CODE: -30041        SQL_STATE: 57013
        SECTION NUM: 00000     STMT NUM: 00060   STMT TYPE: CONNECT TO
        SQLCAID: SQLCA     SQLCABC: 00136   SQLWARN:
        SQLERRD: 0, 0, 0, 0, 0, 198  SQLERRP: TPF0CRDA
        SQLERRML: 0  SQLERRMSG:

        PROGRAM: QXOE      ECB ADDRESS:   6D6000    RDB: DB23PRD
        SQL_CODE: -1024        SQL_STATE: 51007
        SECTION NUM: 00001     STMT NUM: 00102   STMT TYPE: SELECT
        SQLCAID: SQLCA     SQLCABC: 00136   SQLWARN:
        SQLERRD: 0, 0, 0, 0, 0, 217  SQLERRP: TPF0CRDA
        SQLERRML: 0  SQLERRMSG:

        PROGRAM: QXOE      ECB ADDRESS:   6D6000    RDB: DB23PRD
        SQL_CODE: -1024        SQL_STATE: 51007
        SECTION NUM: 00000     STMT NUM: 00079   STMT TYPE: COMMIT
        SQLCAID: SQLCA     SQLCABC: 00136   SQLWARN:
        SQLERRD: 0, 0, 0, 0, 0, 217  SQLERRP: TPF0CRDA
        SQLERRML: 0  SQLERRMSG:

        PROGRAM: QXOE      ECB ADDRESS:   6D6000    RDB: DB23PRD
        SQL_CODE: -1024        SQL_STATE: 51007
        SECTION NUM: 00000     STMT NUM: 00079   STMT TYPE: COMMIT
        SQLCAID: SQLCA     SQLCABC: 00136   SQLWARN:
        SQLERRD: 0, 0, 0, 0, 0, 217  SQLERRP: TPF0CRDA
        SQLERRML: 0  SQLERRMSG:

         END OF TABLE, 4 ENTRIES DISPLAYED
```

Related Information

See the *TPF Application Requester User's Guide* for more information about the structured query language (SQL) and the SQL trace table.

ZSTVS CLOCK–Set System Clocks in STV Mode

Use this command to set the system clocks while you are running the program test vehicle (PTV) in system test vehicle (STV) mode.

Requirements and Restrictions

- You can specify the ADD or SET parameter only when STV mode is active and the TPF system is in NORM state.
- You can specify the RESTORE parameter only when the system is 1052 state. STV does not need to be active.

Format

```
►►──ZSTVS CLOCK── ──┬─┬─ADD──  ──hour──minute──┬─────────────────────────┬─┬──────►◄
                    │ └─SET─┘                  └─ ──month──day──year──┘  │
                    └─RESTORE─────────────────────────────────────────────┘
```

ADD
: adds the specified time and date to the system clock.

SET
: sets the system clock to the specified time and date.

hour
: is the hour from 00–23.

minute
: is the minute from 00–59.

month
: is the month from 01–12.

day
: is the day from 01–31.

year
: is the year from 00–99.

RESTORE
: resets the system clock to its previous time and date.

Additional Information

- You do not need to display the time using the ZDTIM command before you enter this command.
- You can enter this command from the input screen (on the TUT tape) or from a computer room agent set (CRAS) console.
- The STV sends the response to the CRAS console and to the RTL tape.

Examples

The system clock is set to 12:00 in the following example.

```
User:   ZSTVS CLOCK SET 1200

System: ATIM0001I 12.00.55 SUBSYSTEM BSS  LOCAL STANDARD TIME
```

In the following example, 1 minute is added to the system clock.

```
User:   ZSTVS CLOCK ADD 0001

System: ATIM0001I 08.02.11 SUBSYSTEM BSS  LOCAL STANDARD TIME
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSTVS DSPLY–Display Message Block

Use this command to display the current (active) message block being processed and the next (backup) message block to be processed.

Requirements and Restrictions

None.

Format

```
►►──ZSTVS DSPLY──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

The current (active) message block being processed and the next (backup) message block to be processed are displayed in the following example.

```
User:   ZSTVS DSPLY

System: BMS60001I 09.47.24 MSG BLOCKS BEING PROCESSED BY PTV
        002 ACTIVE
        003 BACKUP
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSTVS PAUSE–Pause STV

Use this command to pause the program test vehicle (PTV) when it is running in system test vehicle (STV) mode.

Requirements and Restrictions

PTV must be running in STV mode.

Format

```
►►──ZSTVS PAUSE──┬────────────┬──────────────────────────────────►◄
                 └──BLblock───┘
```

BL*block*
: pauses PTV after all the messages in the specified message block are entered, where *block* is the hexadecimal number of a message block. If you omit this parameter, PTV pauses after the messages in the current message block are entered.

Additional Information

To start PTV again from the message block where it ended, enter the ZSTVS START command and specify the POSIT parameter.

Examples

In the following example, PTV is paused after the messages in the current message block are entered.

```
User:   ZSTVS PAUSE

System: PTVL0001I 08.02.53 LAST MESSAGE BLOCK PROCESSED BL  3
        PAUSE REQUESTED  STV NOW IDLE
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSTVS START–Start STV

Use this command to start the program test vehicle (PTV) when it is running in system test vehicle (STV) mode.

Requirements and Restrictions

- The TPF system must be in NORM state.
- When you run PTV in STV mode, only one test unit is allowed on a test unit tape (TUT). However, a single test unit can span more than one TUT tape.
- If the test unit that you want to run spans more than one TUT tape, specify the BL parameter or the RUNID parameter to start PTV.

Format

```
►►──ZSTVS START── ──┬─BLstart──┬── ──┬─A─┬──number──┬───────────┬──────────────►◄
                    ├─POSIT────┤     └─B─┘          └── ──BLend─┘
                    └─RUNID────┘
```

BL*start*
: starts processing the TUT tape at the specified message block, where *start* is the hexadecimal number of a message block.

POSIT
: restarts PTV from the message block where it ended when PTV was paused.

  Note: Never specify this parameter to restart PTV after a catastrophic error.

RUNID
: starts PTV from the beginning of the TUT tape.

A starts PTV in asynchronous mode.

B starts PTV in burst mode.

number
: is the number of messages, from 1–9, to be maintained in the TPF system in asynchronous mode or to be supplied in each burst.

BL*end*
: pauses PTV at the specified message block on the TUT tape, where *end* is the hexadecimal number of a message block. If you omit this parameter, PTV continues until you enter the ZSTVS PAUSE or ZSTVS STOPT command.

  Note: The value that you specify for BL*end* must be greater than the value that you specify for BL*start*.

Additional Information

None.

Examples

PTV is started in asynchronous mode in the following example. One message is maintained by the TPF system.

```
User:   ZSTVS START RUNID A1

System: COSK0079A 07.48.09 *CP* HPN    MOUNT TUT TAPE FOR INPUT

User:   ZTMNT TUT 425 AI NL

System: COTM0046I 07.49.01 TMNT HPN    TAPE TUT MOUNTED ON DEVICE 425
                           VSN *NONE* G0001 S0001 D38K   NL  NOBLK
        BMS20002I 07.49.01
         RUNID LC001            A1L             PH3 TEST                   MODE A1

        PTVL0001I 07.50.27 LAST MESSAGE BLOCK PROCESSED BL  4
        END OF MESSAGES FROM TAPE  STV NOW IDLE
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSTVS STOPT–STOP STV

Use this command to stop the program test vehicle (PTV) when it is running in system test vehicle (STV) mode. PTV can be either active or paused.

Requirements and Restrictions

- PTV must be running in STV mode.
- Enter this command only after all the messages in the test unit are entered. Otherwise, a catastrophic error can occur.
- After you enter this command, enter the ZSTVS TEST command and specify the RESET parameter.

Format

```
►►──ZSTVS STOPT──────────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

PTV is stopped in the following example.

```
User:   ZSTVS STOPT

System: PTVL0002I 12.02.26 STV STOPPED  DISMOUNT RTL
        COTC0087A 12.02.26 TCLS HPN    REMOVE TUT FROM DEVICE 425
                           VSN *NONE*  NOBLK
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSTVS TEST–Set or Deactivate PTV

Use this command to control the program test vehicle (PTV) testing mode.

Requirements and Restrictions

- The TPF system must be in uniprocessor mode.
- You can specify the YES, STV, or LIVE parameter only when you are prompted by the TPF system during restart.
- Mount a standby tape if output from the PTV test runs is expected to exceed the capacity of one tape reel. When you mount a standby tape, tape switching occurs automatically through the control program.
- When you run PTV in system test vehicle (STV) mode, only one test unit is allowed on a test unit tape (TUT). However, a single test unit can span more than one TUT tape.

Format

```
►►──ZSTVS TEST── ──┬─RESET────┬────────────────────────────────────►◄
                   ├─YES──────┤
                   ├─STV──────┤
                   ├─LIVE─────┤
                   └─END LIVE─┘
```

RESET
resets PTV and allows you to change the PTV testing mode.

Note: You must perform an initial program load (IPL) when you specify this parameter.

YES
runs PTV in phase 3 testing mode.

STV
runs PTV in STV testing mode.

LIVE
ends PTV and returns control to the TPF system.

END LIVE
ends live input from consoles during phase 3 testing mode.

Note: This parameter is valid only when PTV is running in phase 3 testing mode.

Additional Information

None.

Examples

PTV is reset in the following example.

```
User:   ZSTVS TEST RESET

System: PTVV0001I 11.26.37 REMOVE TAPES VIA ZTOFF MSG
        PTV RESET TO INITIATE IPL  ENTER  ZRIPL
```

PTV is started in STV testing mode in the following example.

```
System: PTVB0001I 11.26.41 IDENTIFY TEST RUN
        PH3  - ZSTVS TEST YES
        STV  - ZSTVS TEST STV
        LIVE - ZSTVS TEST LIVE

User:   ZSTVS TEST STV

System: CSMP0099I 11.26.41 010000-B ZSTVS TEST STV
        PTVB0002I 11.26.41 PHASE3 INHIBITED
```

Related Information

See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about PTV.

ZSYSG ALTER–Alter System Generation Option Values

Use this command to change some of the system generation options that are defined using SIP. Only the file copy of the SYSTC bits is changed.

Requirements and Restrictions

You cannot use this command to alter the system generation options on a loader general file.

Format

```
               ┌─,─────────────┐
               ▼               │
►►──ZSYSG Alter───┬────┬──┬─COMXT──┬─┴─────────────────────────►◄
                  └─NO─┘  ├─BPLOAD─┤
                          ├─DBRBUF─┤
                          ├─DLOCK──┤
                          ├─DTAC───┤
                          ├─KPFBK──┤
                          ├─LTRCE──┤
                          ├─SELACT─┤
                          ├─TPFAR──┤
                          ├─UCS1───┤
                          ├─VEQR───┤
                          └─WUNS───┘
```

NO
turns the system generation options off.

COMXT
allows you to call the message router user exit (COBC).

BPLOAD
bypasses loading the core resident program area (CRPA) during restart while the TPF system is cycling to 1052 state. Instead, the core resident programs are loaded when they are called for the first time.

Note: Programs allocated using the SALO PRELOAD parameter are loaded to the CRPA regardless of the value specified for this parameter.

DBRBUF
allows you to write to buffered devices during database reorganization (DBR). See the *TPF Database Reference* for more information.

DLOCK
allows the TPF system to run record hold deadlock detection.

DTAC
includes detached data blocks in the online minidump.

Note: The number of blocks included is defined by DETDATAX in the CZOCP DSECT.

KPFBK
specifies keypoint fallback. See the RAM macro in *TPF System Generation* for more information.

LTRCE
specifies DASD lock tracing. Use this parameter for debugging purposes only.

SELACT
enables or disables the E-type loader selective activate function. When enabled, selected ECBs can enter the programs in a selectively activated loadset. Otherwise, no ECBs can enter the programs in a selectively activated loadset.

TPFAR
allows support for the TPF Application Requester (TPFAR) feature.

UCS1
allows you to call the communication source common user exit (UCS1).

VEQR
specifies migration mode.

WUNS
causes the WTOPC message package to send unsolicited messages to remote terminals. Otherwise, the WTOPC message package sends remote terminals solicited messages.

Additional Information

- You can change only the system generation options listed in the syntax of this command; other system generation options cannot be changed using this command.
- Changes that are made using this command take effect the next time an initial program load (IPL) is performed on the TPF system.
- You can change updatable subsystem-unique system generation bits from any subsystem. Updatable system generation bits that are subsystem-shared can be changed only from the basic subsystem (BSS).
- If an error occurs when you change one or more of the system generation options, none of the options are changed.

Examples

System generation option values are changed in the following example.

```
User:   ZSYSG A NOVEQR

System: SYSG0001I 13.43.41 FILE COPY OF SYSTEM GENERATION BITS UPDATED
        AN IPL MUST BE DONE TO IMPLEMENT THE NEW SYSTEM GENERATION
```

Related Information

- See *TPF System Generation* for more information about SIP.
- See *TPF General Macros* for more information about the SYSTC macro.

ZSYSG DISPLAY–Display System Generation Option Settings

Use this command to display the settings of system generation options that are initially defined using SIP.

Requirements and Restrictions

You can enter this command only from the prime CRAS.

Format

```
              ┌─CORE─┐  ┌─ALL──────┐
►►──ZSYSG Display──┴─FILE─┴──┼──────────┼──────────────────────►◄
                             ├─CP0GENn──┤
                             ├─USRGENn──┤
                             └─option───┘
```

CORE
displays the core copy of the system generation bit settings.

FILE
displays the file copy of the system generation bit settings.

ALL
displays all system generation bit settings.

CP0GEN*n*
displays a list of the valid keywords and their settings for the system generation bit in the CP0GEN byte, where *n* is a decimal number between 0 and 11 that represents the system generation bit.

USRGEN*n*
displays a list of the valid keywords and their settings for the system generation bit in the USRGEN byte, where *n* is a decimal number between 0 and 7 that represents the system generation bit.

option
is a system generation keyword. Specify this parameter as one of the SYSTC tags defined in the SYSTG or SYSUG macro with the first 2 characters removed.

Note: System options are defined in the SYSTG macro and user options are defined in the SYSUG macro.

Additional Information

None.

Examples

The following example displays all of the system generation options.

```
User:   ZSYSG DISPLAY ALL

System: SYSG0004I 13.39.46 CORE COPY OF SYSTEM GENERATION BIT SETTINGS
           BYTE      SETTING       BYTE       SETTING

         *CP0GEN0  B'10111110'    CP0GEN10  B'00000000'
          CP0GEN1  B'01010011'    CP0GEN11  B'01100000'
          CP0GEN2  B'10110000'    USRGEN0   B'00000000'
         *CP0GEN3  B'01101000'    USRGEN1   B'00000000'
          CP0GEN4  B'00111111'    USRGEN2   B'00000000'
         *CP0GEN5  B'00001111'    USRGEN3   B'00000000'
          CP0GEN6  B'00011001'    USRGEN4   B'00000000'
         *CP0GEN7  B'10010000'    USRGEN5   B'00000000'
          CP0GEN8  B'11101111'    USRGEN6   B'00000000'
          CP0GEN9  B'01010010'    USRGEN7   B'00000000'
        * CONTAINS ONE OR MORE SUBSYSTEM UNIQUE BITS
```

The following example displays the system generation options for bit 3 of the CP0GEN byte.

```
User:   ZSYSG DISPLAY CP0GEN3

System: SYSG0005I 13.39.46 CORE COPY OF SYSTEM GENERATION BIT SETTINGS FOR CP0GEN3

        KEYWORD   SETTING   UPDATEABLE

                  RESERVED     NO
        *MAPSP    ON           NO
         MSGRC    ON           NO
                  RESERVED     NO
         NEWGF    ON           NO
        *NFQTK    OFF          NO
                  RESERVED     NO
         DFDDA    OFF          NO
        * DENOTES SUBSYSTEM UNIQUE BIT
```

The following example displays information about the VEQR system generation option.

```
User:   ZSYSG DISPLAY VEQR

System: SYSG0007I 13.39.46 CORE COPY OF SYSTEM GENERATION BIT VEQR IS OFF
         UPDATEABLE, SS SHARED
```

Related Information

- See *TPF System Generation* for more information about SIP.
- See *TPF General Macros* for more information about the SYSTC macro.

ZSYSL–Display or Change Priority Class Shutdown Levels

Use this command to display or change shutdown levels for certain priority classes and block types. These shutdown levels pertain to entry control blocks (ECBs) that issue the LODIC macro.

Requirements and Restrictions

You can only enter this command from the basic subsystem (BSS).

Format

```
►►──ZSYSL── ──┬─Display── ──Class-─┬─BATCH───┬──────────────────────────┬───────────►◄
              │                    ├─LOBATCH─┤                          │
              │                    ├─IBMLO───┤                          │
              │                    └─IBMHI───┘                          │
              │                                ┌──────────────────────┐ │
              │                                ▼                      │ │
              └─Alter── ──Class-─┬─BATCH───┬───┬─,CMB─┬──-value──────┴─┘
                                 ├─LOBATCH─┤   ├─,ECB─┤
                                 ├─IBMLO───┤   ├─,FRM─┤
                                 └─IBMHI───┘   ├─,IOB─┤
                                               └─,SWB─┘
```

Display
displays the shutdown levels for each block type.

Alter
changes the shutdown levels for selected block types.

Class
displays or changes the shutdown level value associated with one of the following priority classes:
- BATCH
- LOBATCH
- IBMLO
- IBMHI

Note: The IBMLO and IBMHI priority classes are reserved for use by IBM.

CMB
changes the shutdown level percentage for common blocks.

ECB
changes the shutdown level percentage for ECBs.

FRM
changes the shutdown level percentage for 4 KB frames.

IOB
changes the shutdown level percentage for input/output (I/O) blocks.

SWB
changes the shutdown level percentage for system work blocks (SWBs).

value
is the new shutdown level percentage from 0–100. This shutdown level

percentage represents the minimum percentage of blocks that must be available to avoid shutdown for a certain transaction classification. For example, SWB-10 means that if the number of available system work blocks (SWBs) falls below 10% of the total number of SWBs, shutdown is in effect for that priority class.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZSYSL HELP
ZSYSL ?

Examples

In the following example, the shutdown levels for 4 KB frames, common blocks, and system work blocks for the LOBATCH priority class are changed.

```
User:   ZSYSL A C-LOBATCH,FRM-55,CMB-60,SWB-75

System: SYSL0003I 10.52.44
        OLD SHUTDOWN LEVELS FOR CLASS LOBATCH

                    TOTAL    SHUTDOWN  SHUTDOWN
        KEYWORD   ALLOCATED    LEVEL    PCT-AGE
          CMB       2186        983       45
          ECB        765        229       30
          FRM       2769       1107       40
          IOB       4480       1568       35
          SWB        512        486       95

        NEW SHUTDOWN LEVELS FOR CLASS LOBATCH

                    TOTAL    SHUTDOWN  SHUTDOWN
        KEYWORD   ALLOCATED    LEVEL    PCT-AGE
          CMB       2186       1311       60
          ECB        765        229       30
          FRM       2769       1522       55
          IOB       4480       1568       35
          SWB        512        384       75
        END OF DISPLAY
```

In the following example, the shutdown levels for the BATCH priority class are displayed.

```
User:   ZSYSL DISPLAY CLASS-BATCH

System: SYSL0002I 17.52.21
        CURRENT SHUTDOWN LEVELS FOR CLASS BATCH

                    TOTAL    SHUTDOWN  SHUTDOWN
        KEYWORD   ALLOCATED    LEVEL    PCT-AGE
          CMB        258        123       48
          ECB        541        259       48
          FRM       2709       1300       48
          IOB       2272       1090       48
          SWB       1128        541       48
        END OF DISPLAY
```

Related Information

See *TPF General Macros* for more information about shutdown levels and their relationship to the LODIC macro.

ZTDEV–Modify and Display Tape Device Status for Automatic Tape Mounting

Use this command to:

- Enable, disable, or resume automatic tape mounting for a tape device.
- Assign or unassign a tape device to or from a tape group.
- Display the tape groups to which a tape device is assigned and the automatic tape mounting status of the tape device.
- Migrate or clear the defined tape device assignment table (DTDAT).

Requirements and Restrictions

- Use the MIGRATE and CLEAR parameters only when prompted by the TPF system during tape restart. The TPF system prompts you to enter the ZTDEV command with the MIGRATE or CLEAR parameter specified if the DTDAT control record is not valid and the ordinal number of the current processor is from 1 to 7. If the ordinal number of the current processor is 0, the TPF system tries to migrate the DTDAT records. If the ordinal number of the current processor is greater than 7, the TPF system automatically initializes the DTDAT records.
- Use the MIGRATE and CLEAR parameters only for the basic subsystem (BSS).

Format

```
>>──ZTDEV── ────────────────────────────────────────────────────────────────►

               ┌─ ──┐
               ↓    │
►──┬─┬─ENable──┬── ──cuu─┴──────────────────────────────────────────────┬─►◄
   │ ├─DISAble─┤                                                        │
   │ └─RESume──┘                                                        │
   ├─DISPlay── ──cuu────────────────────────────────────────────────────┤
   │            ┌─ ──┐                                                  │
   │            ↓    │                                                  │
   ├─ASsign── ──cuu──┴── ──GROUP── ──┬─group─┬── ┌──────────────┐ ──────┤
   │                                 └─ALL───┘   └─ ──SSU ALL───┘       │
   │              ┌─ ──┐                                                │
   │              ↓    │                                                │
   ├─UNAssign── ──cuu──┴── ──GROUP── ──┬─group─┬── ┌───────────────────┐┤
   │                                   └─ALL───┘   └─ ──SSU── ──┬─ALL──┤│
   │                                                            └─ssun─┘│
   ├─MIGrate────────────────────────────────────────────────────────────┤
   └─CLear──────────────────────────────────────────────────────────────┘
```

ENable
enables automatic tape mounting for each specified tape device.

DISAble
disables automatic tape mounting for each specified tape device.

RESume
resumes automatic tape mounting for each specified tape device.

DISPlay
displays the tape groups to which the specified tape device is assigned and the automatic tape mounting status for the tape device.

ASsign
assigns the specified tape devices to the specified tape group. Each tape device is allowed to automatically mount tapes assigned to the specified tape group in the current subsystem user. You can assign each tape device to a maximum of 31 tape groups.

UNAssign
unassigns the specified tape devices from the specified tape group.

cuu
is a 3- to 4-digit hexadecimal tape device address.

GROUP
specifies a tape group.

group
is a 1- to 8-character alphanumeric tape group name.

ALL
assigns the specified tape devices to all tape groups.

SSU ALL
assigns or unassigns the specified tape devices to or from the specified tape group for all the subsystem users.

Note: You can unassign tape devices from all the subsystem users only if you used this parameter to assign the tape devices to all the subsystem users.

SSU *ssun*
unassigns the specified tape devices from an inactive subsystem user, where *ssun* is the name of an inactive subsystem user.

Note: You can specify this parameter only from the basic subsystem (BSS).

MIGrate
migrates the DTDAT records from the ordinal-based, processor unique #TDATR fixed file record type to the file address compute (FACE) program table processor unique #TDTDR fixed file record type. If there is no #TDATR fixed file record or if the DTDAT records in the #TDATR fixed file record are not valid, the DTDAT records in the #TDTDR fixed file record type are cleared and initialized during tape restart.

Notes:

1. The MIGRATE parameter works only on the DTDAT records for the current processor; the DTDAT records for other processors are not migrated.
2. Use the MIGRATE parameter only if the current processor has valid #TDATR fixed file records. (The #TDATR records are valid if the processor was IPLed before tape record migration was applied.) Otherwise, enter the ZTDEV command with the CLEAR parameter specified.

CLear
clears the DTDAT records in the #TDTDR fixed file record type. These records are initialized during tape restart.

Note: The CLEAR parameter works only on the DTDAT records for the current processor; the DTDAT records for other processors are not cleared.

Additional Information

None.

Examples

In the following example, automatic tape mounting is enabled on the specified tape devices.

```
User:   ZTDEV ENABLE 480 481

System: COT50001I 08.36.32 TDEV HPN - ENABLE STATUS AT COMPLETION
        ENABLE 0480 - COMPLETE
        ENABLE 0481 - COMPLETE
        END OF STATUS
```

In the following example, automatic tape mounting is disabled on the specified tape devices.

```
User:   ZTDEV DISABLE 480 481

System: COT50002I 08.37.01 TDEV HPN - DISABLE STATUS AT COMPLETION
        DISABLE 0480 - COMPLETE
        DISABLE 0481 - COMPLETE
        END OF STATUS
```

In the following example, the specified tape device is assigned to the GROUP1 tape group.

```
User:   ZTDEV ASSIGN 482 GROUP GROUP1

System: COT50003I 09.40.50 TDEV HPN - ASSIGN STATUS AT COMPLETION
        ASSIGN 0482 GROUP GROUP1 SSU HPN - COMPLETE
        END OF STATUS
```

In the following example, the specified tape device is unassigned from the GROUP1 tape group.

```
User:   ZTDEV UNASSIGN 482 GROUP GROUP1

System: COT50004I 10.28.38 TDEV HPN - UNASSIGN STATUS AT COMPLETION
        UNASSIGN 0482 GROUP GROUP1 SSU HPN - COMPLETE
        END OF STATUS
```

The automatic tape mounting status for the specified tape device is displayed in the following example along with the names of the tape groups to which the tape device is assigned.

```
User:   ZTDEV DISPLAY 483

System: COT50005I 08.41.12 TDEV HPN - DISPLAY STATUS AT COMPLETION
        DISPLAY 0480 - START OF DISPLAY
           AUTOMOUNT DISABLED
           GROUP      SSN    SSUN
           TEST1      BSS    HPN
        END OF STATUS
```

In the following example, the DTDAT records are migrated from the #TDATR fixed file record type to the #TDTDR fixed file record type.

```
User:   ZTDEV MIGRATE

System: COT50007I 08.36.32 TDEV HPN - MIGRATE STATUS AT COMPLETION
        MIGRATE - COMPLETE
        END OF STATUS
```

In the following example, the DTDAT records in the #TDTDR fixed file record type are cleared.

```
User:   ZTDEV CLEAR

System: COT50008I 08.36.32 TDEV HPN - CLEAR STATUS AT COMPLETION
        CLEAR - COMPLETE
        END OF STATUS
```

Related Information

None.

ZTERM–Display Terminal Information

Use this command to display the following terminal information:
- RCB file address
- WGTA entry address
- Application name
- Logical unit.

Requirements and Restrictions

None.

Format

```
►►──ZTERM── ──┬─lniata─┬──┬──────────┬──────────────────────────►◄
              └─leid───┘  └──cpuid───┘
```

lniata
is a line number, interchange address, and terminal address.

leid
is a logical end-point identifier.

cpuid
is the 1-character alphanumeric CPU ID of a processor.

Additional Information

None.

Examples

Terminal information is displayed for the 4E0000 terminal in the following example.

```
User:   ZTERM 4E0000

System: TERM0001I 19.30.44
         RCB FILE ADDRESS   :  FC02C1DD
         WGTA ENTRY ADDRESS :  00C05514
         APPLICATION NAME   :  NONE
         ASSOCIATED WITH LU :  NONE
```

Related Information

See *TPF Data Communications Services Reference* for more information about the WGTA.

ZTEST–System Test Driver

Use this command to start the CVZZ program.

Requirements and Restrictions

You must change the CVZZ program to accommodate operations in your organization. For example, you can use the CVZZ program to start test drivers. If you do not change the CVZZ program, an error message is displayed.

Format

```
►►──ZTEST──────────────────────────────────────────────────────►◄
```

Additional Information

None.

Examples

None.

Related Information

None.

ZTGRP—Define, Delete, and Display Tape Groups

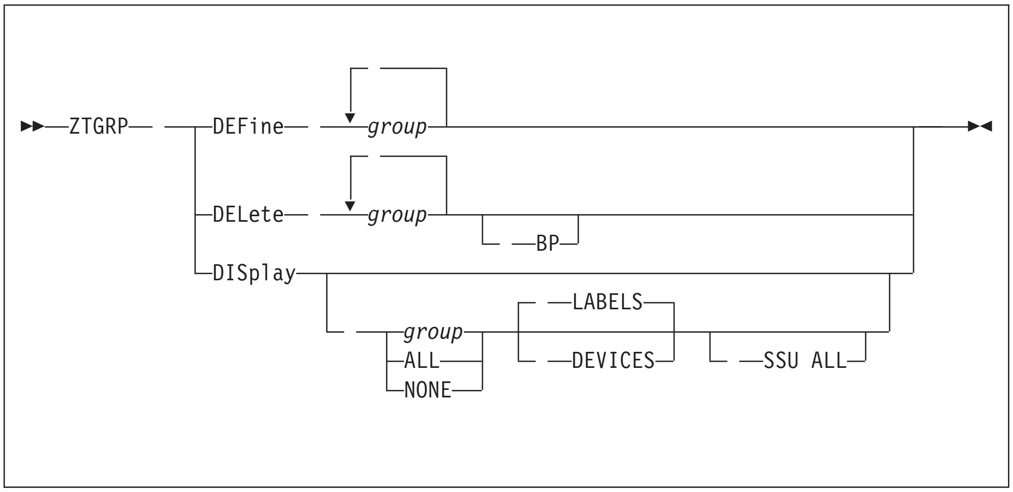
Use this command to:

- Define tape groups
- Delete tape groups
- Display tape groups
- Display a list of tape devices that are assigned to a specific tape group
- Display a list of tape labels that are assigned to a specific tape group.

Requirements and Restrictions

You can define and delete tape groups only from the basic subsystem (BSS).

Format



DEFine
defines a tape group for all subsystem users.

DELete
deletes a tape group from all subsystem users.

Note: You cannot delete a tape group if any tape labels are assigned to that tape group. In addition, if any subsystem user is not currently active, you must specify the BP parameter to delete the tape group.

DISplay
displays information about a tape group.

Note: If you specify only this parameter, information about all the tape groups that are currently defined is displayed.

group
is a 1- to 8-character alphanumeric tape group name.

BP
deletes a tape group when one or more subsystem users are not active.

ALL
displays the tape labels or tape devices that are assigned to the ALL tape group.

NONE
displays the tape labels or tape devices that are assigned to the NONE tape group.

LABELS
displays all the tape labels that are assigned to the specified tape group.

DEVICES
displays all the online and offline tape devices that are assigned to the specified tape group.

SSU ALL
displays all the tape labels or tape devices that are assigned to the specified tape group for all subsystem users. If you omit this parameter, the tape labels and tape devices are displayed for the current subsystem user.

Additional Information

You can define a maximum of 254 tape groups.

Examples

The GROUP1 and GROUP2 tape groups are defined in the following example.

```
User:   ZTGRP DEFINE GROUP1 GROUP2

System: COT70001I 09.40.28 TGRP HPN DEFINE - STATUS AT COMPLETION
        GROUP GROUP1   - DEFINED
        GROUP GROUP2   - DEFINED
        END OF DISPLAY
```

The tape devices assigned to the GROUP1 tape group are displayed in the following example.

```
User:   ZTGRP DISPLAY GROUP1 DEVICES

System: COT70005I 09.42.11 TGRP HPN DISPLAY - DEVICES ASSIGNED TO GROUP GROUP1
        DEVICE ADDRESS   SS NAME    SSU NAME
        0482             BSS        HPN
        END OF DISPLAY
```

The GROUP2 tape group is deleted in the following example. Notice that the BP parameter is required because one or more subsystem users are not currently active.

```
User:   ZTGRP DELETE GROUP2

System: COS70018E 10.17.58 ALL SSUs ARE NOT ACTIVE, BYPASS REQUIRED

User:   ZTGRP DELETE GROUP2 BP

System: COT70002I 10.18.23 TGRP HPN DELETE - STATUS AT COMPLETION
        GROUP GROUP2   - DELETED
        END OF DISPLAY
```

Related Information

None.

ZTHLT–Stop Tracing Selected Records

Use this command to stop the selective file trace (SFT) debugging tool. The SFT debugging tool monitors the updating of selected file addresses during a specified trace period.

Use the SFT debugging tool and the selective file dump (SFD) debugging tool (also referred to as the selective file dump and trace (SFDT) debugging tools) to locate file-related errors during online operations or while testing under the control of the program test vehicle (PTV) utility.

Requirements and Restrictions

None.

Format

```
►►──ZTHLT──────────────────────────────────────────────────────────►◄
```

Additional Information

All output from the SFDT debugging tools is written to the real-time tape (RTL/RTA). SFD and SFT output may be interspersed with other test or control program output also on the real-time tape. Use the offline diagnostic output formatter (DOF) utility to process the SFDT output. See *TPF Program Development Support Reference* for examples of the SFDT output.

Examples

The following example starts and stops the SFT debugging tool. This example monitors the specified file addresses and logs the file addresses for any records that were updated during the trace period but not specified for monitoring.

```
User:   ZTRCE A0 184800B3/18480033/184800A3

System: TRCE INITIATED

User:   ZTHLT

System: TRCE COMPLETED
```

Related Information

- See “ZTRCE–Trace Selected Records” on page 1387 for more information about starting the SFT debugging tool.
- See “ZSELD–Dump Selected Records” on page 1251 for more information about the SFD debugging tool.
- See *TPF Program Development Support Reference* for more information about the selective file dump and trace (SFDT) debugging tools and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline diagnostic output formatter (DOF) utility.

- See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about the PTV utility.

ZTICL–Emergency Tape Removal

Use this command to release and optionally make available an item in the tape status table (TSTB) that was left seized by a software error. The ID of the ECB-controlled tape support program that seized the TSTB item is displayed.

Requirements and Restrictions

- You can enter this command only from the basic subsystem (BSS).
- This command is provided only to recover from a tape control program failure. Indiscriminate use can seriously impair system integrity.

Format

```
►►──ZTICL── ──cuu──┬─────┬──────────────────────────────────────────────►◄
                   └─ ──F┘
```

cuu
: is a 3-digit hexadecimal tape device address.

F
: resets the TSTB item to available (that is, any tape mounted on the specified tape device is logically dismounted).

Additional Information

None.

Examples

In the following example, the specified tape device is made available after it was seized by the COTS program.

```
User:   ZTICL 482 F

System: COTY0123I 10.18.23 TICL BSS DEVICE 482 UNSEIZED AND FREED - SEIZING
System: PROGRAM - COTS
```

Related Information

None.

ZTINT–Initialize Tape

Use this command to initialize a tape. The standard volume label (VOL1), dummy header label (HDR1), and tapemark required at the beginning of an output tape is created.

Requirements and Restrictions

Enhanced capacity cartridge system tape cartridges can be used only on tape devices that can process 3480-2 XF format tapes.

Format

```
►►──ZTINT── ──cuu──┬──────────┬──┬────────────┬──┬───────┬──────────────►◄
                   └─ ──volume─┘  ├─ ──D38K────┤  ├─ ──BP─┤
                                  ├─ ──D38K2───┤  └─ ──NT─┘
                                  ├─ ──F38K────┤
                                  ├─ ──F38K2───┤
                                  └─ ──F16X128─┘
```

cuu
is a 3-digit hexadecimal tape device address.

volume
is a 6-character alphanumeric volume serial number. If you omit this parameter, the existing volume serial number is read and rewritten.

D38K
see F38K parameter.

D38K2
see F38K2 parameter.

F38K
writes the header labels in a format with an operating density of 38K bytes per inch (Bpi).

F38K2
writes the header labels in a format with an operating density of 38K Bpi in 3480-2 XF format.

F16X128
writes the header labels in a format of 16 tracks per pass and 128 total tracks.

BP
allows you to initialize an unexpired tape.

Attention: Use this parameter with caution because it can destroy valid data.

NT
allows you to initialize tapes that were never previously recorded or that were recorded at a density that is not compatible with the specified tape device.

Attention: Use this parameter with caution because it can destroy valid data.

Additional Information

If you do not specify a tape format for writing the header labels (for example, the F38K parameter or the F16X128 parameter) the format defaults to the highest level supported by the specified tape device.

Examples

In the following example, the tape that is on the specified tape device is initialized.

```
User:   ZTINT 481 F38K2

System: COTI0304I 10.31.19 TINT HPN    DEVICE 481 INITIALIZATION COMPLETE
                           VSN A00046 FORMAT  38K2
```

Related Information

None.

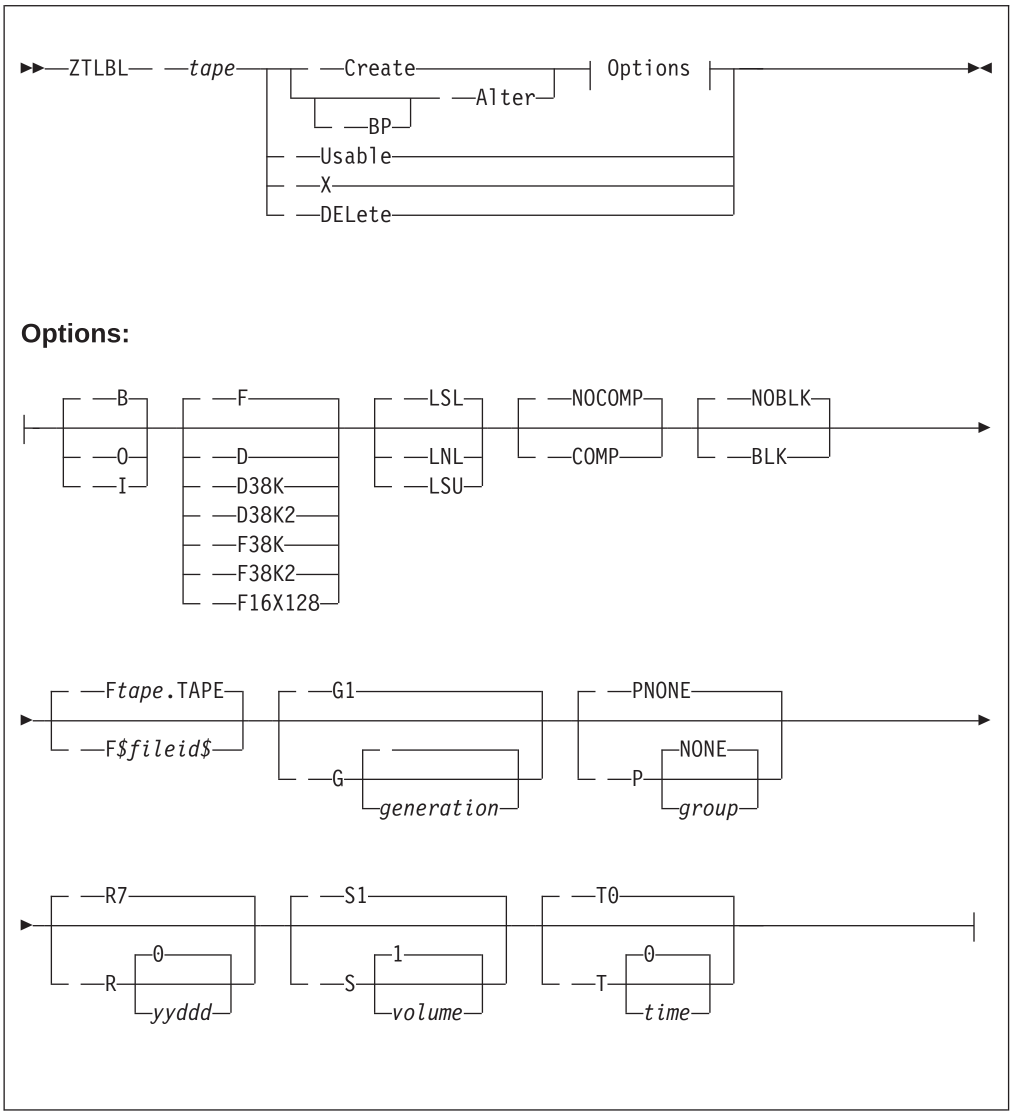
ZTLBL–Tape Label Maintenance

Use this command to create, change, or delete a tape label, or to assign a tape label to a tape group.

Requirements and Restrictions

- The tape label directory records and the tape label mask records must be initialized before you can create any tape labels. Enter the ZTLMR command to initialize these records.
- You must create a tape label for each tape name that you want to use in the TPF system, except ALT.

Format



tape
: is a 3-character tape name. The first 2 characters are alphabetic and the third character is alphanumeric.

Create
: creates a new tape label.

Note: You cannot use the new tape label until you enter this command with the USABLE parameter.

BP
changes the tape label information while the tape is mounted.

Attention: If you specify this parameter, offline support systems may not be able to read the tape. In addition, it may be difficult for the TPF system to read the tape.

Alter
changes the options defined for a tape label.

Note: If you did not specify the BP parameter, you cannot use the changed tape label until you enter this command with the USABLE parameter.

Usable
allows you to use a recently created or changed tape label.

X or DELete
removes a tape label from the TPF system.

Note: You cannot remove a tape label if a tape is mounted that is using that tape label.

B allows the tape to be used for both input and output.

O allows the tape to be used only for output.

I allows the tape to be used only for input.

F allows the tape to be read or written in any format.

D allows the tape to be read or written at any density.

D38K
see F38K parameter.

D38K2
see F38K2 parameter.

F38K
allows the tape to be read or written in a format with an operating density of 38K bytes per inch (Bpi).

F38K2
allows the tape to be read or written in a format with an operating density of 38K Bpi in 3480-2 XF format.

F16X128
allows the tape to be read or written in a format of 16 tracks per pass and 128 total tracks.

LSL
specifies that the tape has standard header labels.

LNL
specifies that the tape does not have standard header labels if it is used for input. (Output tapes always have standard header labels.)

LSU
specifies that the tape has standard header labels followed by user labels. Use the appropriate user exit to create the user labels.

COMP
specifies that the output tape is written in compacted mode.

NOCOMP
specifies that the output tape is written in uncompacted mode.

BLK
specifies that the tape is written or read in blocked mode.

NOBLK
specifies that the tape is written or read in unblocked mode.

F*$fileid$*
is a file or data set identifier for the tape, where *fileid* is the 1- to 17-character alphanumeric identifier and *$* is the delimiter, which can be any character except the following:
- A character contained in the identifier itself
- Blank
- Comma (,)
- Right parenthesis ( ))
- Equal sign (=)
- Hyphen (-)
- Plus sign (+)
- Period (.)

G*generation*
is the next generation of the tape to be read or written. You can omit leading zeros.

Note: If you create a tape for output, the generation number in the TLMR is incremented after the tape is closed and no longer matches the generation number on the tape. If you then attempt to mount the tape as input, using the same symbolic name, you will receive a generation number conflict. To avoid this, do not to use generation numbers for the file or use different symbolic names for input and output phases. Otherwise, you may have to reset the generation number manually and check it before each tape is mounted.

P assigns the tape label to the specified tape group.

group
is the 1- to 8-character alphanumeric tape group name.

NONE
does not assign the tape label to any tape groups.

R*yyddd*
specifies the retention period of the tape, where *yy* is the number of years and *ddd* is the number of days. You can omit leading zeros.

S*volume*
specifies the volume sequence number for the next volume of the tape to be read or written, where *volume* is the volume sequence number. You can omit leading zeros.

T*time*
specifies the time interval for synchronizing the buffer when the tape is mounted in block mode, where *time* is the time interval from 0–255 seconds. You can omit leading zeros.

Note: A time interval of 0 specifies that no time-initiated buffer synchronization is performed.

Additional Information

- You can also change the options that are defined for the tape label when you enter the ZTMNT command to mount the tape.
- The number of tape labels maintained by the TPF system depends on the number of tape label directory records and tape label mask records that are defined in the TPF system.

Examples

In the following example, the INP tape label is created for unlabeled input tapes.

```
User:   ZTLBL INP C I LNL

System: COTL0002I 09.53.05 DTLB BSS  - TAPE LABEL INFORMATION
           INP - NOT USABLE    I/O-INPUT   L-NL   FMT-ALL  NOCOMP
           NOBLK               T  - 000
```

The RTB tape label is changed in the following example. It is now a tape label for output tapes with standard header labels.

```
User:   ZTLBL RTB A O LSL

System: COTL0002I 10.03.05 DTLB BSS  - TAPE LABEL INFORMATION
           RTB - NOT USABLE    I/O-OUTPUT  L-SL   FMT-ALL  NOCOMP
           NOBLK               T  - 000
           F-RTB.TAPE          G  -0001    S-0001
           RETENTION PERIOD -  00007
           LAST MOUNTED     -
           LAST FILE SERIAL -
           GROUP            -  NONE
```

The changed RTB tape label is made usable in the following example.

```
User:   ZTLBL RTB U

System: COTJ0313I 10.02.18 TLBL BSS    COMPLETE - TAPE RTB LABEL USABLE
```

The RTB tape label is deleted in the following example.

```
User:   ZTLBL RTB X

System: COTJ0314I 09.54.24 TLBL BSS    COMPLETE - TAPE RTB LABEL DELETED
```

Related Information

None.

ZTLMR–Initialize Tape Label Records

Use this command to initialize the tape label directory records and the tape label mask records.

Requirements and Restrictions

- Specify the DEACTIVATE parameter to deactivate the tape label directory records and tape label mask records before you initialize them.
- In a multiple database function (MDBF) environment, you must initialize the tape label directory records and tape label mask records for each subsystem user.
- If you change the number of tape label directory records or tape label mask records in an MDBF environment, you must initialize them again for all subsystem users in the complex.

Format

```
►►──ZTLMR── ──┬─INITialize─┬──────────────────────────────────────────►◄
              ├─DEACtivate─┤
              └─ACTivate───┘
```

INITialize
initializes the tape label directory records and tape label mask records, and then activates them.

Note: You must deactivate the records before you can initialize them.

DEACtivate
deactivates the tape label directory records and tape label mask records.

ACTivate
activates the tape label directory records and tape label mask records that were inadvertently deactivated.

Additional Information

None.

Examples

The tape label directory records and tape label mask records are initialized in the following example. Notice that the records must be deactivated first.

```
User:   ZTLMR INIT

System: COTH0092E 06.56.54 TLMR HPN    TPLD RECORD ACTIVE - NO INITIALIZATION

User:   ZTLMR DEAC

System: COTH0044I 06.56.54 TLMR HPN    COMPLETE

User:   ZTLMR INIT

System: COTH0044I 06.56.54 TLMR HPN    COMPLETE
```

Related Information

None.

ZTMNT–Mount a Tape

Use this command to mount a tape. When you mount a tape, a symbolic tape name and the associated tape label are assigned to a physical tape device. The header labels for the tape on the tape device are created or checked as appropriate.

Requirements and Restrictions

- You cannot mount a standby tape without a corresponding active tape.
- When mounting the first tape volume of a multiple-volume input file, ensure that the information in the header label agrees with the corresponding information in the tape label. (That is, do not specify the BP parameter to mount the tape.) If this information does not agree, you cannot switch to a previous tape volume (backward tape switch). You can, however, use this command to change the options that are defined in the tape label, if necessary.

Format

```
►►──ZTMNT── ──tape── ──cuu──┬─ ──A─┬──┬──────────┬──┬───────────┬──────────────────────────►
                            └─ ──S─┘  ├─ ──I─────┤  ├─ ──SL─────┤
                                      └─ ──O─────┘  ├─ ──NL─────┤
                                                    └─ ──BP─────┘

►──┬────────────────────┬──┬─────────────┬──┬────────────┬──┬──────────────┬───────────►◄
   └─ ────┬─D38K─────┬──┘  ├─ ──COMP─────┤  ├─ ──BLK─────┤  └─ ──OVerlay───┘
          ├─D38K2────┤     └─ ──NOCOMP───┘  └─ ──NOBLK───┘
          ├─F38K─────┤
          ├─F38K2────┤
          └─F16X128──┘
```

tape
: is a 3-character tape name. The first 2 characters are alphabetic and the 3rd character is alphanumeric.

  Note: You cannot specify a tape name of ALL, XXX, or YYY.

cuu
: is a 3-digit hexadecimal tape device address.

A mounts the tape as an active tape.

S mounts the tape as a standby tape.

  Note: A corresponding active tape must exist before you can mount a standby tape.

I mounts the tape as an input tape.

O mounts the tape as an output tape.

SL
: mounts an input tape when it has standard header labels but the tape label indicates that it has no standard header labels.

NL
mounts an input tape when it has no standard header labels but the tape label indicates that it has standard header labels.

BP
allows you to mount an output tape before it expires, to mount an input tape when the tape label and standard header labels conflict, or to mount an input tape that has a VOL1 label but no HDR1 label.

D38K
see F38K parameter.

D38K2
see F38K2 parameter.

F38K
allows the tape to be read or written in a format with an operating density of 38K bytes per inch (Bpi).

F38K2
allows the tape to be read or written in a format with an operating density of 38K Bpi in 3480-2 XF format.

F16X128
allows the tape to be read or written in a format of 16 tracks per pass and 128 total tracks.

COMP
specifies that the output tape is written in compacted mode.

Note: This parameter is not valid for input, standby, or alternate (ALT) tapes, or when mounting a tape on a tape device where the improved data recording capability (IDRC) feature is not installed or enabled.

NOCOMP
specifies that the output tape is written in uncompacted mode.

Note: This parameter is not valid for input, standby, or ALT tapes.

BLK
specifies that the tape is written or read in blocked mode.

Notes:

1. This parameter is not valid when mounting standby or ALT tapes.
2. You must mount RCP tapes as BLK tapes.

NOBLK
specifies that the tape is written or read in unblocked mode.

Note: This parameter is not valid when mounting standby or ALT tapes.

OVerlay
converts the ALT tape that is mounted on the specified tape device to the specified tape name.

Note: If an error occurs while converting the ALT tape, the ALT tape is dismounted.

Additional Information

- The tape options that are currently defined in the tape label are applied to the tape unless you specify new values for these options when you enter this command.
- The special tape name ALT designates a common standby output tape. You can mount ALT tapes in place of standard output tapes as long as the tape densities are the same. You must mount ALT tapes as active output tapes. You can mount any number of ALT tapes.

Examples

The RTB tape is mounted on the specified tape device in the following example.

```
User:   ZTMNT RTB 483

System: COTM0310I 10.05.05 TMNT BSS    TAPE RTB MOUNTED ON DEVICE 483
                           VSN A00150 G0001 S0001 F38K   SL  NOBLK  NOCOMP
```

The BP parameter is required to mount the RTC output tape in the following example because the RTC tape has not expired.

```
User:   ZTMNT RTC 483 BP

System: COTM0008W 10.14.17 TMNT BSS    DEVICE 483 VSN A00150
                           UNEXPIRED FILE OVERWRITTEN
        COTM0310I 10.14.17 TMNT BSS    TAPE RTC MOUNTED ON DEVICE 483
                           VSN A00150 G0003 S0001 F38K   SL  NOBLK  NOCOMP
```

Related Information

None.

ZTMSL–Display, Change, Add, or Remove Time Slice Attributes

Use this command to display, change, or remove time slice attributes, or to add a time slice name with its associated attributes to the time slice name table.

Requirements and Restrictions

You can only enter this command from the basic subsystem (BSS).

Format

```
►►──ZTMSL── ──────────────────────────────────────────────────────────►

►──┬─Display──┬──────────┬──────────────────────────────────────┬─────►
   │          └─ ──name──┘                                      │
   ├─Remove── ──name────────────────────────────────────────────┤
   │                  ┌────────────────┐                        │
   │                  ▼                │                        │
   ├─Change── ──name──┬─,MAXEcb-maxe──┬┴────────────────────────┤
   │                  ├─,MAXTime-maxt─┤                         │
   │                  ├─,MInsusp-mins─┤                         │
   │                  └─,Runtime-runt─┘                         │
   └─Add── ──name── ──MAXEcb-maxe,MAXTime-maxt,MInsusp-mins,Runtime-runt─┘

   ┌─ ──Copy-File───────┐
►──┼────────────────────┼──────────────────────────────────────────►◄
   └─ ──Copy-──┬─Core─┬─┘
               └─Both─┘
```

Display
: displays the attributes associated with a time slice name if a time slice name was specified. Otherwise, a list of all supported time slice names is displayed.

Remove
: removes a time slice name from the time slice name table. ECBs that are referencing the deleted time slice name are unaffected.

Change
: Changes the attributes defined for a specific time slice name.

Add
: adds a new time slice name.

name
: is a 1- to 8-character alphanumeric time slice name.

MAXEcb-*maxe*
: assigns the maximum number of ECBs for the specified time slice name, where *maxe* is a value from 0–9999.

MAXTime-*maxt*
: assigns the maximum amount of accumulated run time that an ECB enabled for time slicing can use before exiting with an error, where *maxt* is a value from 0–9999999 milliseconds (ms). A value of 0 means that there is no predefined maximum run time; the ECB will run until it completes its function.

MInsusp-*mins*
assigns the minimum amount of time that an ECB enabled for time slicing can remain suspended, where *mins* is a value from 0–9999 ms.

Runtime-*runt*
assigns the maximum amount of time that an ECB enabled for time slicing can run without giving up control, where *runt* is a value from 10–500 ms.

Note: This value cannot exceed the SIP generated value for the application timeout counter (set to 500 ms by IBM).

Copy
selects from which copy of the time slice name table to display, change, add, or remove time slice attributes, where:

File
specifies the file copy.

Core
specifies the core copy.

Both
specifies both the file and core copies.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTMSL HELP
  ZTMSL ?
- The ZTMSL command will update the core copy of the time slice name table only on the processor where the command was entered. You must enter the command again on the other loosely coupled processors, or an initial program load must be performed on those processors to ensure that the core copy of the time slice name table is identical across a loosely coupled system.
- When using the ZTMSL command to remove an IBM time slice name (defined in segment CTMS), or to change its attributes, the same corresponding change must be reflected in segment CTMS. If CTMS is not updated the name or original attributes will appear again when you IPL the processor.
- The following three hardcoded time slice names are reserved for use by IBM:
  – IBMLOPRI
  – IBMHIPRI
  – IBMINDEF

Examples

The values of the MAXECB, MINSUSP, MAXTIME, and RUNTIME parameters for the BIGSORT time slice name are changed in the file copy of the time slice name table in the following example.

```
User:   ZTMSL C BIGSORT MAXE-20,MI-40,MAXT-5000,R-200

System: TMSL0005I 10.52.44
        OLD TIME SLICE ATTRIBUTES FOR NAME BIGSORT ON FILE

            MAXECB-15    MAXTIME-2000      MINSUSP-80     RUNTIME-100

        NEW TIME SLICE ATTRIBUTES FOR NAME BIGSORT ON FILE

            MAXECB-20    MAXTIME-5000      MINSUSP-40     RUNTIME-200

        END OF DISPLAY
```

The supported time slice names are displayed from the core copy of the time slice name table in the following example.

```
User:   ZTMSL D C-C

System: TMSL0002I 10.52.44
        EXISTING TIME SLICE NAMES IN CORE

            IBMRSVD1       IBMRSVD2        CLOCKCHG        GLOBALU
            SCHEDCHG       FAREWAR         QUOTEME         BIGSORT
            LITTLSRT

        END OF DISPLAY
```

The BSEARCH time slice name is added to both the file copy and the core copy of the time slice name table in the following example.

```
User:   ZTMSL A BSEARCH MAXE-20,MI-40,MAXT-5000,R-200 C-B

System: TMSL0004I 10.52.44
        NEW TIME SLICE ATTRIBUTES FOR NAME BSEARCH ON FILE

            MAXECB-15    MAXTIME-2000      MINSUSP-80     RUNTIME-100

        NEW TIME SLICE ATTRIBUTES FOR NAME BSEARCH IN CORE

            MAXECB-15    MAXTIME-2000      MINSUSP-80     RUNTIME-100
        END OF DISPLAY
```

The OLDSORT time slice name is removed from the core copy of the time slice name table in the following example.

```
User:   ZTMSL R OLDSORT C-C

System: TMSL0010I 10.52.44
        THE TIME SLICE NAME OLDSORT HAS BEEN REMOVED IN CORE
```

Related Information

See *TPF System Macros* for more information about time slice names and attributes and their relation to the TMSLC macro.

ZTOCU–Dismount Tapes by Logical Control Unit

Use this command to dismount all tapes from a specified logical control unit.

Requirements and Restrictions

- You can enter this command only from the basic subsystem (BSS).
- The active system error dump tape is not dismounted from the control unit.

Format

```
►►──ZTOCU── ──cu────────────────────────────────────────────────────────────►◄
```

cu is the 2-digit hexadecimal address of a control unit.

Additional Information

Enter the ZTOFF command to dismount the active system error dump tape.

Examples

All tapes except the active system error dump tape are dismounted from the specified control unit in the following example:

```
User:   ZTOCU 48

System: COTG0087A 12.55.36 TOCU HPN    REMOVE KMP FROM DEVICE 481
                           VSN A00050  NOBLK  COMP
        COS10102I 12.55.36 TOCU BSS    CONTROL UNIT 48 - DISMOUNT COMPLETE
```

Related Information

None.

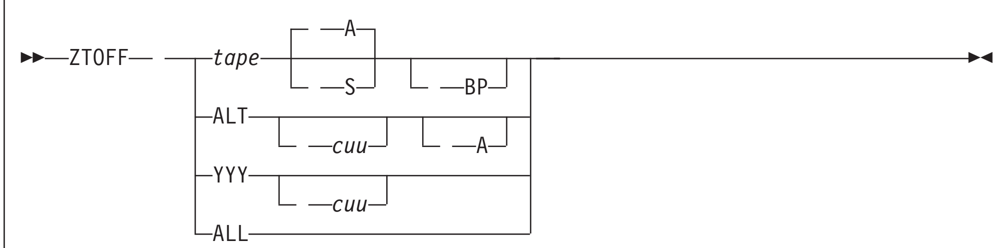
ZTOFF–Dismount, Rewind, and Unload Tape

Use this command to:
- Dismount a tape
- Dismount an alternate (ALT) tape on a specified tape device
- Rewind and unload the tape on a specified tape device
- Dismount all of the mounted tapes from the TPF system.

Requirements and Restrictions

- You can dismount RTA and RTL tapes only when the TPF system is in 1052 state.
- If an active tape has a corresponding standby tape, you must dismount the standby tape first.

Format



tape
is a 3-character tape name. The first 2 characters are alphabetic and the 3rd character is alphanumeric.

A dismounts an active tape.

S dismounts a standby tape.

BP
dismounts a tape that is currently being used by an application (that is, an open active general tape), or dismounts a tape when a real-time tape switch is in progress.

Attention: Use this parameter with caution because it causes the ECB that is using the specified tape to exit with a system error.

ALT
dismounts the ALT tape on the specified tape device.

cuu
is a 3-character hexadecimal tape device address. If you do not specify a tape device address for the ALT parameter, the first ALT tape that is located in the tape status table (TSTB) is dismounted.

YYY
rewinds and unloads the tape on the specified tape device.

ALL
dismounts all the tapes in the TPF system, except the system error dump tape.

Note: You can specify this parameter only from the basic subsystem (BSS).

Additional Information

Volume statistics are not logged automatically to the system log tape when you cycle down the TPF system in preparation for a machine switchover or other initial program load (IPL). To ensure that these statistics are not lost, enter **ZTOFF ALL** to dismount all the tapes and log the volume statistics before you perform a machine switchover or IPL.

Examples

The RTB tape is dismounted in the following example.

```
User:   ZTOFF RTB

System: COTG0300A 12.12.47 TOFF BSS    REMOVE RTB FROM DEVICE 484
                           VSN A00095 G0002 S0001 F38K   SL  NOBLK  NOCOMP
```

All the tapes in the TPF system except the system error dump tape are dismounted in the following example.

```
User:   ZTOFF ALL

System: COTG0087A 12.55.36 TOFF HPN    REMOVE KMP FROM DEVICE 481
                           VSN A00050  NOBLK  COMP
        COTF0114I 12.55.36 TOFF BSS    ALL TAPES DISMOUNTED EXCEPT DUMP TAPE
```

Related Information

None.

ZTPLD–Activate Auxiliary Loader

Use this command to load program data from a general data set, tape, virtual reader, or user-defined device to an image using the auxiliary loader.

Requirements and Restrictions

- You must disable the image before you can load program data to it.
- Before you can enter this command, you must assign a data definition name to the input device using the ZDSMG DEFINE or ZDSMG MT command.
- If you are loading loadsets from a tape input device, you must mount the tape input device using the ZTMNT command.
- If you are loading program data from a virtual reader input device, you can load only one file from the virtual reader at a time. Therefore, you cannot load program data from an input device that was defined as VRDR-ALL.

Format

```
►►──ZTPLD── ──image── ──ddname──────────────────────────────────────►◄
```

image
: is the 5- to 8-character alphanumeric name of the image to which you want to load the program data.

ddname
: is the 1- to 16-character alphanumeric data definition name of the input device (general data set, tape, virtual reader, or user-defined device) that contains the program data.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTPLD HELP
  ZTPLD ?
- When loading keypoints, the auxiliary loader first clears the keypoint staging area for the image, and then loads the keypoints to the keypoint staging area. Therefore, the keypoint staging area always contains only the keypoints that were last loaded to the image. The auxiliary loader does ***not*** clear the keypoint staging area if no keypoints are being loaded. You can use the ZIMAG KEYPT command to move the keypoints from the keypoint staging area to the working keypoint area.

Examples

Data is loaded from a virtual reader to the NEWCP image in the following example.

```
User:   ZTPLD NEWCP VRDR

System: TPLD0001I 19.39.59 STARTING LOAD TO IMAGE NEWCP FROM DDNAME VRDR
        TPLD0002I 19.39.59 CPS0JA LOADED
        TPLD0004I 19.39.59 LOAD COMPLETE
```

Related Information

- See *TPF System Installation Support Reference* for more information about the auxiliary loader.
- See *TPF System Installation Support Reference* and *TPF System Generation* for more information about TPF images.

ZTPLF FILL–Load All Tapes in a Tape Library Category

Use this command to continuously load all the tapes in a tape library category one after the other to a specified tape device using the tape library dataserver. After the first tape is rewound and unloaded, the tape library dataserver automatically loads the next tape in the tape library category. This process is continued until there are no more tapes in the tape library category.

Requirements and Restrictions

None.

Format

```
                                       ┌─ ──NOSEquence─┐
►►──ZTPLF FILL── ──DEV-cuu── ──CAT-catg─┼───────────────┼──────────────────►◄
                                       └─ ──SEquence───┘
```

DEV-*cuu*
: is the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

CAT-*catg*
: is a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

SEquence
: loads the tapes in the order that they were added to the tape library category.

  Note: This parameter is valid only if the specified tape library category is not also being loaded to another tape device.

NOSEquence
: loads the tapes in the order that is most efficient for the tape library dataserver.

Additional Information

- Remember that a tape is automatically removed from its current tape category when you mount or unload that tape. Therefore, if a tape in the specified tape library category is mounted on or unloaded from another tape device, it is automatically removed from that tape library category at that time and will not be loaded on the specified tape device. This strategy allows you to continuously load a tape library category using one or more tape devices.
- If the tape library dataserver is continuously loading the tapes in a tape library category, it preloads the tapes to the automatic cartridge loader (ACL) if the ACL is installed and a tape is currently loaded in the tape device. Otherwise, the tape library dataserver loads the tapes directly to the tape device when the tape device is available.

Examples

In the following example, the tape library dataserver loaded the first tape in the X'1000' tape library category to the specified tape device. After this tape is rewound and unloaded, the tape library dataserver will automatically load the next tape in the tape library category.

```
User:   ZTPLF FILL DEV-F30 CAT-1000

System: CORD0210I 08.13.11 TPLF HPN DEV 0F30 - FILL CATEGORY 1000
        CORE0234I 08.13.11 TPLF BSS    VSN ALJT07 AUTO-LOADED ON F30 - CAT 1000
```

Related Information

None.

ZTPLF LOAD–Load a Tape

Use this command to load a tape to a specified tape device using the tape library dataserver.

Requirements and Restrictions

You cannot enter this command if a tape is already loaded on the tape device.

Format

```
                              ┌─  ──────────┐
                              ▼             │
►►──ZTPLF LOAD── ──DEV-cuu── ──┬─VSN-volume─┬┴──────────────────────────►◄
                               └─CAT-catg───┘
```

DEV-*cuu*
: is the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

VSN-*volume*
: loads the specified tape volume, where *volume* is a 6-character alphanumeric volume serial number.

CAT-*catg*
: loads a tape from the specified tape library category, where *catg* is a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

Additional Information

None.

Examples

In the following example, the ALJT07 tape in the X'1000' tape library category is loaded to the specified tape device.

```
User:   ZTPLF LOAD DEV-F30 VSN-ALJT07 CAT-1000

System: CORD0205I 08.13.11 TPLF HPN VSN ALJT07 LOADED ON 0F30 FROM CAT-1000
```

Related Information

None.

ZTPLF MOVE–Reassign the Tape Library Category of a Tape

Use this command to move a tape from one tape library category to another.

Requirements and Restrictions

None.

Format

```
►►──ZTPLF MOVE── ──DEV-cuu── ──VSN-volume── ──FROM-catg── ──TO-catg──────────►◄
```

DEV-*cuu*
is the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

VSN-*volume*
is the 6-character alphanumeric volume serial number of a tape.

FROM
specifies the tape library category from which you want to move the tape.

TO
specifies the tape library category to which you want to move the tape.

catg
is a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

Additional Information

None.

Examples

The ALJT07 tape is moved from the X'1000' tape library category to the X'3000' tape library category in the following example.

```
User:   ZTPLF MOVE DEV-F30 VSN-ALJT07 FROM-1000 TO-3000

System: CORD0226I 08.13.11 TPLF HPN VSN ALJT07 MOVED FROM 1000 TO 3000
```

Related Information

None.

ZTPLF QUERY–Query Status

Use this command to display the following information:
- Status of a tape device
- Status of a tape
- List of tapes that were added to a specific tape library category
- List of tape library categories that are reserved.

Requirements and Restrictions

None.

Format

```
►►──ZTPLF QUERY ─DEV-cuu──┬───────────────────────────────────────────────────┬──►◄
                          │                ┌─NOSORT─┐                         │
                          ├─ ─CAT-catg─────┤        ├───┬─────────────────┬───┤
                          │                └─SORT───┘   └─ ─SEQ-number────┘   │
                          ├─ ─REserved-─┬─hostname─┬──┬──────────────────┬────┤
                          │             └─*────────┘  └─ ─SEQ-catg───────┘    │
                          └─ ─VSN-volume──────────────────────────────────────┘
```

DEV-*cuu*
displays the status of the specified tape device, where *cuu* is the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

CAT-*catg*
displays the tapes that were added to the specified tape library category, where *catg* is a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

NOSORT
displays the tapes in the order that they were added to the tape library category.

SORT
displays the tapes in alphabetic order if fewer than 600 tapes were added to the tape library category.

SEQ-*number*
displays the tapes that were added to the specified tape library category starting at a specific sequence number, where *number* is a 1- to 8-digit hexadecimal number.

When you specify the CAT parameter, the display lists up to 590 tapes. If there are more than 590 tapes, the display ends with the category sequence number of the last tape volume displayed. Use this parameter to specify that sequence number to see the remaining volumes in the category.

Note: You can use the sequence numbers only to determine the relative position of a volume in a category to other volumes in the category. The sequence number of the last volume added to the category cannot be used to calculate how many volumes are in the category and the first volume in the category may not have a sequence number of one.

REserved
displays a list of the tape library categories that were reserved for the specified host system.

hostname
is the 1- to 8-character alphanumeric token name used by a host system.

Note: The host system is not required to be a TPF system.

*****
displays a list of the tape library categories that were reserved for the current host system.

SEQ-*catg*
begins the list at the specified tape library category, where *catg* is a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

VSN-*volume*
displays the status of the specified tape volume, where *volume* is the 6-character alphanumeric volume serial number of the tape.

Additional Information

None.

Examples

The status of the F30 tape device is displayed in the following example.

```
User:   ZTPLF QUERY DEV-F30

System: CORD0227I 08.13.11 TPLF HPN STATUS FOR DEVICE 0F30
        VSN-ALJT07  CAT-1000  FILL-AUTO  ACL-EMPTY
```

A list of the tapes that were added to the X'1000' tape library category for the tape library dataserver that is attached to the specified tape device is displayed in the following example. The tapes are listed in the order that they were added to the tape library category.

```
User:   ZTPLF QUERY DEV-F30 CAT-1000

System: CORD0228I 08.13.11 TPLF HPN 000026 VOLUMES FOR CATEGORY 1000
        VOLUMES UNSORTED
        ALJL02  ALJL07  ALJT10  ALJT07  ALJL05  ALJT08  ALJT04  ALJT05
        ALJL04  ALJT09  ALJL03  AL3091  ALJL09  AL3100  ALJL08  AL3101
        ALJL06  AL3099  AL3098  AL3089  AL3090  AL3093  AL3097  ALJT01
        AL3219  AL3218
        END OF DISPLAY
```

The status of the ALJT07 tape is displayed in the following example.

```
User:   ZTPLF QUERY DEV-F30 VSN-ALJT07

System: CSMP0097I 08.13.11 CPU-B SS-BSS  SSU-HPN  IS-01
        CORD0230I 08.13.11 TPLF HPN STATUS FOR VOLUME ALJT07
        CAT-1000   STATUS- LOADED
```

Related Information

None.

ZTPLF RELEASE–Release a Tape Library Category

Use this command to return a previously reserved tape library category to the library manager so that it is available to other host systems.

Requirements and Restrictions

You cannot release a tape library category if there are tapes added to it.

Format

```
►►──ZTPLF RELEASE── ──DEV-cuu── ──CAT-catg──┬──────────────────────┬──────────►◄
                                            └─ REserved-hostname ──┘
```

DEV-*cuu*
specifies the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

CAT-*catg*
specifies a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

REserved-*hostname*
specifies the 1- to 8-character alphanumeric token name used by the host system when the category specified was reserved.

Note: The host system is not required to be a TPF system.

Additional Information

Use the ZTPLF RESERVE command to reserve a tape library category.

Examples

The X'0110' tape library category is released in the following example.

```
User:   ZTPLF RELEASE DEV-800 CAT-0110

System: CORD223I 09.43.44 TPLF HPN    CATEGORY 0110 RELEASED
```

Related Information

None.

ZTPLF RESERVE–Reserve a Tape Library Category

Use this command to reserve a tape library category. The library manager provides you with a tape library category that has no tapes added to it and that was not already reserved by another host system.

Requirements and Restrictions

It is your responsibility to monitor the use of the tape library categories that you reserve.

Format

```
►►──ZTPLF RESERVE── ──DEV-cuu──┬───────────┬──────────────────────────►◄
                               └─ CAT-catg─┘
```

DEV-*cuu*
: specifies the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

CAT-*catg*
: specifies a 4-digit hexadecimal tape library category from X'0000'–X'FFFF'. See “Tape Library Categories” on page 23 for a description of the different tape library categories.

Additional Information

- Other host systems can use a tape library category that you reserved for the TPF system. For example, other host systems can add tapes to a reserved tape library category, load the tapes in a reserved tape library category, and unload the tapes in a reserved tape library category.
- Enter the ZTPLF RELEASE command to return a tape library category to the library manager so that it is available to other host systems.

Examples

In the following example, a tape library category is reserved for the tape library dataserver that is attached to the specified tape device.

```
User:   ZTPLF RESERVE DEV-800

System: CORD218I 15.12.38 TPLF HPN    CAT 0114 RESERVED FOR BTPFNET
```

Related Information

None.

ZTPLF UNLOAD–Unload a Tape

Use this command to unload tapes from the specified tape device using the tape library dataserver. You can unload the current tape or all the tapes in a tape library category.

Requirements and Restrictions

If the tape that is currently loaded in the tape device is mounted, enter the ZTOFF command to dismount the tape before you enter this command.

Format

```
                               ┌─ ──NOALL─┐
►►──ZTPLF UNLOAD── ──DEV-cuu───┼──────────┼──────────────────────────────►◄
                               └─ ──ALL───┘
```

DEV-*cuu*
: is the 3- to 4-digit hexadecimal address of a tape device that is attached to the tape library dataserver.

NOALL
: unloads the current tape from the specified tape device.

  Note: If you previously entered the ZTPLF FILL command to continuously load the tape library category in the tape device, this parameter unloads the current tape from the tape device and then automatically loads the next tape in the tape library category.

ALL
: unloads the current tape from the specified tape device and cancels the continuous load of the other tapes in that tape library category.

  Note: If you try to unload a tape library category without first dismounting the tape that is currently loaded in the tape device, the tape library dataserver cancels the continuous load of the tapes in that tape library category but ***does not*** unload the tape that is currently loaded in the tape device.

Additional Information

None.

Examples

In the following example, the tape that is currently loaded in the specified tape device is unloaded.

```
User:   ZTPLF UNLOAD DEV-F30

System: CORD0214I 08.13.11 TPLF HPN VSN ALJT07 REMOVED FROM 0F30
```

ZTPLF UNLOAD

In the following example, the tape library category that is being continuously loaded in the specified tape device is unloaded. That is, the current tape is unloaded and the continuous load of the other tapes in the tape library category is canceled.

```
User:   ZTPLF UNLOAD DEV-F30 ALL

System: CORD0215I 08.13.11 TPLF HPN FILL CAT 1000 DELETED FROM 0F30
                              ALL VOLUMES REMOVED FROM ACL
                              VSN ALJT10 REMOVED FROM DEVICE
```

Related Information

None.

ZTPSW–Perform a Manual Tape Switch

Use this command to switch an output tape to the next volume.

Requirements and Restrictions

You can enter this command only for output tapes.

Format

```
►►──ZTPSW── ──tape──────────────────────────────────────────────►◄
```

tape
is a 3-character tape name. The first 2 characters are alphabetic and the 3rd character is alphanumeric.

Additional Information

None.

Examples

In the following example, the GEN output tape is switched to the next volume.

```
User:   ZTPSW GEN

System: COTS0382I 10.59.46 TPSW BSS    TAPE GEN SWITCHED FROM 482 TO 483
                           VSN IS NOW A00137
        COTS0087A 10.59.46 TPSW BSS    REMOVE GEN FROM DEVICE 482
                           VSN A00136 G0001 S0001 D38K   SL  NOBLK  COMP
```

Related Information

None.

ZTRAC–Start Macro Tracing for the Real-Time Trace Utility

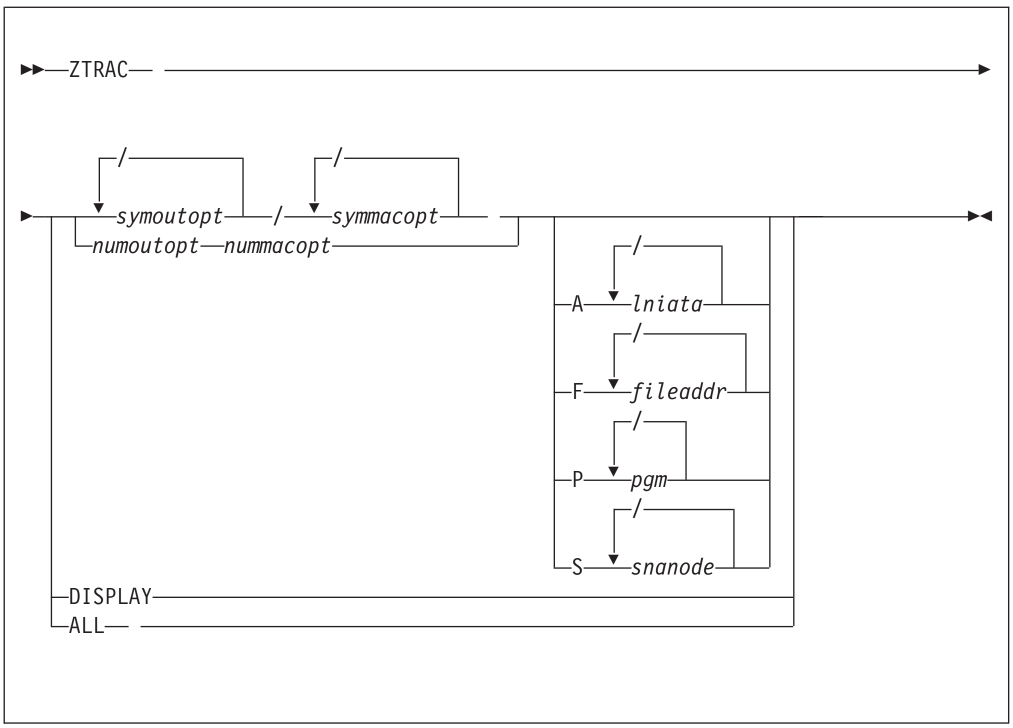
Use this command to start the macro tracing function of the real-time trace (RTT) utility. The RTT utility counts or traces the use of specific macros and related system activity. When you enter the ZTRAC command, the RTT utility traces any macros, including fast wait and fast link (non SVC) macros. The amount of trace data collected is determined by the parameters specified.

Use the ZCNTM command to start the macro counting function. Use the ZSTOP command to stop the macro tracing or macro counting function.

Requirements and Restrictions

Each ZTRAC command must have a corresponding ZSTOP command. If you enter a second ZTRAC command before the count period is ended with a ZSTOP command, the second ZTRAC command is rejected.

Format



symoutopt
is the symbolic name for the output trace option. Specify one or more of the following:

MIN
specifies the minimum output, which includes the:
- Macro name and parameters
- Address of the macro in a program
- Contents of register 8 (R8) and R9
- Condition code at the time the macro was processed
- Terminal address.

REGS
includes the application registers in the output.

ECBW
includes the entry control block (ECB) work area in the output.

ECBR
includes the ECB contents relevant to the macro group in the output. If you specify this value, the output generated for each macro varies with each macro. See Table 13 on page 1380 for more information about the additional output generated for each macro.

CORE
includes the main storage blocks attached to the ECB in the output. If you specify this value, the output generated for each macro varies with each macro. See Table 13 on page 1380 for more information about the additional output generated for each macro.

USER
includes the user work area (expanded ECB) in the output.

symmacopt
is the symbolic name for the macro trace option. Specify one or more of the following:

ENBK
traces enter-type macros and the BACKC macro.

CRET
traces the DEFRC and DLAYC, and create-type macros.

CTRL
traces the EXITC, LMONC, MONTC, and WAITC macros.

FILE
traces file-type macros and the UNFRC macros.

FIND
traces find-type macros.

DATA
traces the FLIPC, GETCC, and RELCC macros.

POOL
traces get file storage-type macros and the RELFC macro.

SEND
traces the CRASC macro and send-type macros.

RTAP
traces the real-time tape macros.

GTAP
traces the AMSSC and FDCTC macros, and the general tape macros.

MISC
traces miscellaneous macros.

SON
traces SON macros.

numoutopt
is the numeric value for the output trace option. Specify one or a combination of the following:

00 specifies the minimum output, which includes the:
- Macro name and parameters

- Address of the macro in a program
- Contents of register 8 (R8) and R9
- Condition code at the time the macro was processed
- Terminal address.

01 includes the application registers in the output.

02 includes the ECB work area in the output.

04 includes the ECB contents relevant to the macro group in the output. If you specify this value, the output generated for each macro varies with each macro. See Table 13 on page 1380 for more information about the additional output generated for each macro.

08 includes the main storage blocks attached to the ECB in the output. If you specify this value, the output generated for each macro varies with each macro. See Table 13 on page 1380 for more information about the additional output generated for each macro.

10 includes the user work area (expanded ECB) in the output.

1F includes all output options in the output.

You can combine numeric output options; that is, you can add the values through binary arithmetic and enter that sum for the trace output value. For example, if you want to collect the ECB work area (02) and the ECB areas relevant to the macro (04), specify 06 for the output option.

nummacopt
is the numeric value for the macro trace option. Specify one of the following:

0 traces SXSVC macros.

1 traces enter-type macros and the BACKC macro.

2 traces the DEFRC and DLAYC, and create-type macros.

3 traces the EXITC, LMONC, MONTC, and WAITC macros.

4 traces file-type macros and the UNFRC macros.

5 traces find-type macros.

6 traces the FLIPC, GETCC, and RELCC macros.

7 traces get file storage-type macros and the RELFC macro.

8 traces the CRASC macro and send-type macros.

9 traces the real-time tape macros.

A traces the AMSSC and FDCTC macros, and the general tape macros.

B traces miscellaneous macros.

D traces SON macros.

F traces all macros.

You ***cannot*** combine numeric macro options. If you want to specify more than one macro option for a command, use the symbolic macro option values.

A*lniata*
specifies the A trace mode. The A trace mode limits the RTT utility to activity assoiated with messages from specified terminals, where *lniata* is the line number, interchange address, and terminal address (LNIATA). If you specify this

parameter, the RTT utility is activated whenever the specified macro is issued by a program activated as a result of a message from one of the identified terminals. Keep the following in mind when specifying the value for *lniata*:

- Specify 2, 4, or 6 hexadecimal characters. A 2-character value represents a line number for LNIATAs. A 4-character value represents a line/interchange for LNIATAs. A 6-character value represents the entire LNIATA.
- You can specify multiple LNIATAs on the same command (separated by the / character), but the LNIATAs must all have the same length. SLC link numbers may be specified as a line number, right-justified, and padded by zeros to meet the 4-character or 6-character length requirement.
- You can specify synchronous data link control (SDLC) pseudo LNIATAs in the same way as actual LNIATAs. You can specify locally attached 3270s in the same way as line numbers. The IA and TA are both X'00'. You can specify SDLC pseudo line numbers for SDLC SNA devices, which must be 6 characters long. The pseudo LNIATA is used to refer to an SDLC SNA device (network control program (NCP), line, cluster controller, or logical unit).
- References can be made to different devices on the same command.
- SDLC pseudo line numbers can to intermixed with LNIATAs on the same command. The SDLC pseudo line numbers must be 6 characters long. The LNIATAs can be 2, 4, or 6 characters long. Any SDLC pseudo line numbers specified are not counted in the maximum number of LNIATAs that can be specified.
- If you do not specify particular address parameters, you can use the ALL parameter to trace input from all terminal addresses (including SDLC pseudo lines).

F*fileaddr*
specifies the F trace mode. The F trace mode limits the RTT utility to activity associated with specified file addresses, where *fileaddr* is a 4- or 8-byte file address. If you specify this parameter, the RTT utility is activated whenever file-type macros use the identified file addresses. You can specify the ALL parameter to trace all ECBs that reference file addresses and these file addresses can be 4- or 8-byte file addresses. You can specify multiple file addresses (separated by the / character); however, the specified file addresses must have the same length: all 4- or 8-byte addresses.

P*pgm*
specifies the P trace mode. The P trace mode limits the RTT utility to activity of the macros issued by specified programs, where *pgm* is the 1- to 4-character name of the program to be traced or counted. If you specify this parameter, the RTT utility is activated whenever the specified macro is issued by the identified program. You can specify multiple program names (separated by the / character) on one command; however, all the names must be the same length: all 1, 2, 3, or 4 characters.

S*snanode*
specifies the S trace mode. The S trace mode limits the RTT utility to activity associated with messages from specified SNA node names, where *snanode* is the 1- to 8-character node name of the SNA logical unit (LU). Keep the following in mind when specifying the value for *snanode*:

- The node names must be associated with the CPU running the RTT utility.
- You can specify multiple SNA node names on the same command (separated by the / character), but length of the command, including the action code and end-of-message (EOM) character does not exceed the device limit.

- You can use the ALL parameter to trace all ECBs generated by SNA input or output.

DISPLAY
displays all symbolic parameters.

ALL
specifies all output options and all macro options. This is the same as entering **ZTRAC 1FF**, using the numeric values 1F for all output options and F for all macro options.

Notes:

1. You must leave a blank space after the ALL parameter for the command to process correctly.
2. Use caution when specifying the amount of trace output to be collected, especially in an online environment. If you request a trace of all output for all macro activity for all ECBs (ZTRAC ALL or ZTRAC 1FF), storage capacity and printout resources may be immediately depleted as the RTT utility tries to collect the maximum trace output for every macro process. The TPF system issues the AB0007 system error when this occurs. See *Messages (System Error and Offline)* for more information about the AB0007 system error.

Additional Information

- The output generated for each macro when you specify the ECBR, 04, CORE, or 08 values varies with each macro. Table 13 shows the additional output generated for each macro.

  Note: ECBR and CORE are the symbolic parameter values while 04 and 08 are the numeric parameter value.

Table 13. Trace Output

Macro Name	RTT Group	ECBR or 04 Output				CORE or 08 Output
	Symbolic/ Numeric	**Entire ECB**	**CBRWs**	**FARWs & FWs**	**None**	
ALASC	ENBK/	X				
AMSSC	GTAP/A				X	
ATTAC	DATA/6		X	X		
BACKC	ENBK/1	X				
CALOSC						
CCIDC	MISC/B				X	
CEBIC	MISC/B	X				
CENVC	MISC/B				X	
CIFRC	MISC/B				X	
CINFC	MISC/B				X	
CIOSC	CTRL/3	X				
CIOUC	SEND/8				X	
CONKC	MISC/B				X	
CRASC	SEND/8		X			
CRATC	MISC/B				X	

Table 13. Trace Output (continued)

Macro Name	RTT Group	ECBR or 04 Output				CORE or 08 Output
	Symbolic/ Numeric	Entire ECB	CBRWs	FARWs & FWs	None	
CREDC	CRET/2				X	
CREEC	CRET/2				X	
CREMC	CRET/2				X	
CRESC	CRET/2				X	
CRETCM	CRET/2				X	
CRETCS	CRET/2				X	
CREXC	CRET/2				X	
CROSC	MISC/B	X				
CSONC	SON/D		X	X		
DEFRC	CRET/2				X	
DETAC	DATA/6		X	X		
DLAYC	CRET/2				X	
DPROC	MISC/B		X	X		
DUMPC	MISC/B				X	
ELLEG					X	
ENTDC	ENBK/1	X				
ENTNC	ENBK/1	X				
ENTRC	ENBK/1	X				
EVINC				X		
EVNTC	CRET/2	X				
EVNWC	CRET/2	X				
EXITC	CTRL/3	X				
FDCTC	GTAP/A	X				
FILEC	FILE/4		X	X		Data Blocks
FILNC	FILE/4		X	X		Data Blocks
FILSCD	FILE/4		X	X		Data Blocks
FILSCP	FILE/4		X	X		Data Blocks
FILUC	FILE/4		X	X		Data Blocks
FINDC	FIND/5		X	X		
FINHC	FIND/5		X	X		
FINSCD	FIND/5		X	X		
FINSCP	FIND/5		X	X		
FINWC	FIND/5	X				Data Blocks
FIWHC	FIND/5	X				Data Blocks
FLIPC	DATA/6		X	X		
FLSPC	FILE/4		X	X		Data Blocks
FLVFC					X	

Table 13. Trace Output (continued)

Macro Name	RTT Group	ECBR or 04 Output				CORE or 08 Output
	Symbolic/ Numeric	Entire ECB	CBRWs	FARWs & FWs	None	
FNSPC	FIND/5		X	X		
FREEC					X	
FSTIC	SON/D				X	
GDSCC					X	
GDSNC	FIND/5		X	X		
GDSRC	FIND/5		X	X		
GETCC	DATA/6		X			
GETFC	POOL/7		X	X		
GETPC	MISC/B		X	X		
GFSCC	POOL/7		X			
GLBUC	MISC/B		X	X		
HASHC	MISC/B				X	
IGATC					X	
INQRC	MISC/B				X	
KEYCC	MISC/B				X	
KEYRC	MISC/B				X	
KEYUC	MISC/B				X	
KLCDC	SON/D		X	X		
KLCLC	SON/D		X	X		
KLCSC	SON/D		X	X		
KSCDC	SON/D		X	X		
KSCLC	SON/D		X	X		
KSCSC	SON/D		X	X		
LMONC	CTRL/3				X	
MAxBC					X	
MALOC					X	
MONTC	CTRL/3				X	
MONWC	CTRL/3	X				
MPIFC					X	
MSPIC				X		
NUMBC					X	
PAUSC	CTRL/3	X				
PLONC1	SEND/8		X			
PLONC2	SEND/8		X			Data Blocks
PKEYC					X	
PHYBC					X	
POLLC	SEND/8				X	

Table 13. Trace Output (continued)

Macro Name	RTT Group	ECBR or 04 Output				CORE or 08 Output
	Symbolic/ Numeric	Entire ECB	CBRWs	FARWs & FWs	None	
POSTC	CRET/2	X				
PRLNC	MISC/B		X			
PROGC				X		
RALOC					X	
RDCDC	MISC/B		X			
RDCTC	MISC/B		X	X		
RELCC	DATA/6		X			
RELFC	POOL/7		X	X		
RELPC	MISC/B		X	X		
REQSC	SEND/8				X	
RIDCC	MISC/B				X	
ROUTC	MISC/8		X			Data Blocks
RTCUC	MISC/B		X	X		
RVTCC	MISC/B				X	
SANSC	SEND/8		X			Data Blocks
SCDCCM	SEND/8		X			Data Blocks
SCDCCS	SEND/8		X			Data Blocks
SENDCA	SEND/8		X			Data Blocks
SENDCB	SEND/8		X			Data Blocks
SENDCC	SEND/8		X			Data Blocks
SENDCK	SEND/8		X			Data Blocks
SENDCL	SEND/8		X			Data Blocks
SENDCT	SEND/8		X			Data Blocks
SETCC	PTV/C	X				
SICFC				X		
SIPCC	SEND/8		X	X		
SKIPT	PTV/C	X				
SLMTC	SEND/8		X			Data Blocks
SNDLC	SEND/8		X			Data Blocks
SNIAC	SEND/8		X			Data Blocks
SONIC	SON/D		X	X		
SOUTC	SEND/8		X			Data Blocks
STLUC	SEND/8		X			Data Blocks
STPMT	PTV/C	X				
STPPT	PTV/C	X				
STXTC	SEND/8		X			Data Blocks
SWISC	ENBK/1	X				

Table 13. Trace Output (continued)

Macro Name	RTT Group	ECBR or 04 Output				CORE or 08 Output
	Symbolic/ Numeric	Entire ECB	CBRWs	FARWs & FWs	None	
SYNCC	SEND/8		X	X		
TASBC	MISC/B				X	
TASNC	GTAP/A				X	
TASTC	MISC/B				X	
TBSPC	GTAP/A		X	X		
TCLSC	GTAP/A				X	
TDCTC	GTAP/A	X				
TDSPC	GTAP/A		X	X		
TDTAC	GTAP/A		X	X		
TIMEC	MISC/B				X	
TOPNC	GTAP/A				X	
TOURC	RTAP/9		X			
TOUTC	RTAP/9		X	X		
TPCNC	GTAP/A		X	X		
TPINC	GTAP/A				X	
TPRDC	GTAP/A		X	X		
TREWC	GTAP/A				X	
TRPMT	PTV/C	X				
TRPPT	PTV/C	X				
TRSVC	GTAP/A				X	
TSYNC					X	
TWRTC	GTAP/A		X			
UATBC	MISC/B	X				
UNFRC	FILE/4		X	X		Data Blocks
URCTC	MISC/B		X	X		
USURC	MISC/B				X	
UXCMC					X	
WAITC	CTRL/3	X				Data Blocks
WTOPC	SEND/8				X	Data Blocks

- If you do not specify a trace mode parameter, the RTT utility is activated each time the seleted macro type is processed. Be aware that this can place excessive demands on TPF system resources and can exhaust available storage.
- All output from the RTT utility is written to the real-time tape (RTL/RTA). Use the offline diagnostic output formatter (DOF) utility to process the RTT output.

Examples

In the following examples tracing occurs when an ECB communicates with SLC link number 4A, and with terminals 3B attached to the pseudo SLC line number 50, interchange 01. The first example uses the symbolic parameters and the second example uses the numeric parameter. These are examples using the A trace mode.

```
User:   ZTRAC ECBR/REGS/ENBK A00004A/50013B

System: RTT IN
```

```
User:   ZTRAC 051A00004A/50013B

System: RTT IN
```

The trace output for these examples is:

- The minimum RTT output, which includes:
  - Macro name and parameters
  - Address of the macro in a program
  - Contents of register 8 (R8) and R9
  - Condition code at the time the macro was processed
  - Terminal address.
- The contents of the application registers
- The relevant parts of the ECB (provided for BACKC and all enter-type macros used by an entry communicating on these lines).

In the following examples one specified file record is traced. The first example uses the symbolic parameters and the second example uses the numeric parameter. These are examples using the F trace mode.

```
User:   ZTRAC CORE/FILE F020C0311

System: RTT IN
```

```
User:   ZTRAC 084F020C0311

System: RTT IN
```

The trace output for these examples is:

- The minimum RTT output, which includes:
  - Macro name and parameters
  - Address of the macro in a program
  - Contents of register 8 (R8) and R9
  - Condition code at the time the macro was processed
  - Terminal address.
- The main storage blocks attached to the ECB, including data event control blocks (DECBs)
- A copy of the record file for all file-type and UNFRC macros for all entries that reference the specified file address.

In the following examples programs beginning with the names ABC, EFC, and EFG are traced. The first example uses the symbolic parameters and the second example uses the numeric parameter. These are examples using the P trace mode.

```
User:   ZTRAC REGS/SEND PABC/EFC/EFG

System: RTT IN
```

```
User:   ZTRAC 018PABC/EFC/EFG

System: RTT IN
```

The trace output for these examples is:

- The minimum RTT output, which includes:
  - Macro name and parameters
  - Address of the macro in a program
  - Contents of register 8 (R8) and R9
  - Condition code at the time the macro was processed
  - Terminal address.
- The contents of the application registers (generated when a CRASC, ROUTC, or send-type macro is issued in the segments with names that begin with ABC, EFC, or EFG).

In the following examples tracing occurs on all entries generated from the single LU with a node name of DEVICE1 or DEVICE2. The first example uses the symbolic parameters and the second example uses the numeric parameter. These are examples using the S trace mode.

```
User:   ZTRAC CORE/CTRL SDEVICE1/DEVICE2

System: RTT IN
```

```
User:   ZTRAC 083SDEVICE1/DEVICE2

System: RTT IN
```

The trace output for these examples is:

- The minimum RTT output generated for the EXITC, WAITC, MONTC, and LMONC macros
- A copy of all main storage blocks attached to the ECB at I/O completion for the WAITC macro, including DECBs.

Related Information

- See “ZCNTM–Start Macro Counting for the Real-Time Trace Utility” on page 224 for more information about starting the macro counting function.
- See “ZSTOP–Stop Macro Tracing or Macro Counting for the Real-Time Trace Utility” on page 1310 for more information about stopping the RTT utility.
- See *TPF Program Development Support Reference* for more information about the RTT utility and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline DOF utility.

ZTRCE–Trace Selected Records

Use this command to start the selective file trace (SFT) debugging tool. The SFT debugging tool monitors the updating of selected file addresses during a specified trace period.

Use the SFT debugging tool and the selective file dump (SFD) debugging tool (also referred to as the selective file dump and trace (SFDT) debugging tools) to locate file-related errors during online operations or while testing under the control of the program test vehicle (PTV) utility.

Requirements and Restrictions

None.

Format

```
                          ┌─/──────────┐
                          ▼            │
►►──ZTRCE── ──┬─Aoption── ────hexfileaddr┴────────────────┬──────►◄
              │                ┌─/─────────────────────────┐ │
              │                ▼                           │ │
              └─┬─Noption─┬── ────rectype*firstord-lastord*┴─┘
                └─Soption─┘
```

A indicates that the file address is expressed as a file address reference format (FARF) record type. Chained records will contain a FARF file address in the forward chain field.

N indicates that the file address is expressed as a file address compute (FACE) program record type and ordinal number. Chained records will contain a FACE-type ordinal in the forward chain field.

S indicates that the file address is a FACE record type and ordinal number. Chained records will contain a FARF file address in the forward chain field.

option
is one of the following dump options:

0 logs file addresses of all file records that were updated during the trace period but not specified for monitoring. This parameter can help you check for unanticipated file activity. If you specify 0, you must specify at least one file address.

1 does not log file addresses.

hexfileaddr
is the file address of the record specified as an 8- or 16-digit hexadecimal value. You can specify up to 82 file addresses.

*rectype******firstord***-***lastord******
is the file address specified as one or more sets of records. Each set consists of a series of file records of one FACE record type with contiguous FACE ordinal numbers, specified with a 19- or 39-byte field, where:

rectype
is the FACE record type indicator in hexadecimal characters.

firstord
is the 6- or 16-digit hexadecimal ordinal number of the first record in the set.

lastord
is the 6- or 16-digit hexadecimal ordinal number of the last record in the set.

You can specify up to 43 fields with one command. The number of records specified in a single command cannot exceed 420.

Additional Information

All output from the SFDT debugging tools is written to the real-time tape (RTL/RTA). SFD and SFT output may be interspersed with other test or control program output also on the real-time tape. Use the offline diagnostic output formatter (DOF) utility to process the SFDT output. See *TPF Program Development Support Reference* for examples of the SFDT output.

Examples

The following example starts and stops the SFT debugging tool. This example monitors the specified file addresses and logs the file addresses for any records that were updated during the trace period but not specified for monitoring.

```
User:   ZTRCE A0 184800B3/18480033/184800A3

System: TRCE INITIATED

User:   ZTHLT

System: TRCE COMPLETED
```

The following example monitors the specified file addresses and does not log the file addresses.

```
User:   ZTRCE N1 00C5*000000-000012*/0099*000000-000004*

System: TRCE INITIATED

User:   ZTHLT

System: TRCE COMPLETED
```

Related Information

- See “ZTHLT–Stop Tracing Selected Records” on page 1342 for more information about stopping the SFT debugging tool.
- See “ZSELD–Dump Selected Records” on page 1251 for more information about the SFD debugging tool.
- See *TPF Program Development Support Reference* for more information about the selective file dump and trace (SFDT) debugging tools and for examples of the output.
- See “Diagnostic Output Formatter” on page 1437 and *TPF Program Development Support Reference* for more information about the offline diagnostic output formatter (DOF) utility.
- See “Program Test Vehicle” on page 1453 and *TPF Program Development Support Reference* for more information about the PTV utility.

ZTRMT–Remount Tape

Use this command to remount a previous volume of an input tape.

Requirements and Restrictions

- You can enter this command only when you are prompted by the TPF system.
- The TPF system must be in 1052 state or higher.
- You can enter this command only for input tapes.

Format

```
►►──ZTRMT── ──tape── ──cuu──────────────────────────────────────────────►◄
```

tape
is a 3-character tape name. The first 2 characters are alphabetic and the 3rd character is alphanumeric.

cuu
is the 3-digit hexadecimal address of the tape device where you want to mount the previous volume.

Additional Information

None.

Examples

In the following example, the RTB input tape is switched to the previous volume on the 483 tape device.

```
User:   ZTRMT RTB 483

System: COTR0118I 11.05.46 TRMT BSS    TAPE RTB SWITCHED BACKWARDS FROM 482 TO 483
        COTR0087A 11.05.46 TRMT BSS    REMOVE RTB FROM DEVICE 482
                           VSN A00136 G0001 S0001 D38K   SL  NOBLK  COMP
```

Related Information

None.

ZTRTE–Manage IP Routing Tables

Use this command to do the following:

- Add Internet Protocol (IP) routing table entries that map either a local IP address or an address of a next-hop IP router to a remote IP address or subnet of remote IP addresses.
- Delete IP routing table entries.
- Modify IP routing table entries.
- Display IP routing table contents.
- Display IP routing table statistics.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).
- You can enter this command only if the IP routing table has been defined by specifying the MAXRTE parameter of the SNAKEY macro as a value greater than 0.
- You can enter this command only if Transmission Control Protocol/Internet Protocol (TCP/IP) native stack support is defined.

Format

```
>>──ZTRTE── ──────────────────────────────────────────────────────────────>

                                                ┌─ ──NETmask-255.255.255.255─┐
>──┬─ADD── ──RIP-ripaddr──┬─ ──LIP-lipaddr───┬──┼────────────────────────────┼──┬──><
   │                      └─ ──NEXThop-nextip┘  └─ ──NETmask-netmask─────────┘  │
   ├─DELete── ──RIP-ripaddr──┬─────────────────────┬────────────────────────────┤
   │                         ├─ ──LIP-lipaddr──────┤                            │
   │                         └─ ──NEXThop-nextip───┘                            │
   ├─DELete── ──┬─ALL──────────────┬────────────────────────────────────────────┤
   │            ├─LIP-lipaddr──────┤                                            │
   │            └─NEXThop-nextip───┘                                            │
   ├─MODify── ──┬─RIP-ripaddr── ──LIP-lipaddr── ──NEXTLIP-newlip─┬──────────────┤
   │            └─LIP-lipaddr── ──NEWLIP-newlip──────────────────┘              │
   ├─DISplay── ──┬─ALL──────────────────────────────────────────┬───────────────┤
   │             │               ┌─ ──NETmask-255.255.255.255─┐ │               │
   │             ├─RIP-ripaddr───┼────────────────────────────┼─┤               │
   │             │               ├─ ──NETmask-netmask─────────┤ │               │
   │             │               └─ ──LIP-lipaddr─────────────┘ │               │
   │             ├─LIP-lipaddr──────────────────────────────────┤               │
   │             └─NEXThop-nextip───────────────────────────────┘               │
   └─STAT───────────────────────────────────────────────────────────────────────┘
```

ADD
: adds a new IP routing table entry. This entry includes a remote IP address or subnet of remote IP addresses, and either a corresponding local IP address or the address of the next-hop IP router used to reach the remote IP address or subnet of remote IP addresses.

RIP-*ripaddr*
: specifies the remote IP address or subnet of remote IP addresses, where *ripaddr* is the numeric remote IP address or subnet of remote IP addresses.

LIP-*lipaddr*
specifies the local IP address, where *lipaddr* is the numeric IP address.

NEXThop-*nextip*
specifies the IP address of the next-hop IP router, where *nextip* is the numeric IP address.

NETmask-*netmask*
specifies the network mask used with the remote IP address supplied in the RIP parameter to determine the subnet of possible remote IP addresses, where *netmask* is the network mask in numeric format.

DELete
deletes IP routing table entries.

ALL
deletes or displays all entries in the IP routing table.

MODify
modifies IP routing table entries. You can also specify a new local IP address by using the NEWLIP parameter.

NEWLIP-*newlip*
specifies the local IP address placed in the selected IP routing table entries, where *newlip* is the numeric IP address.

DISPlay
displays IP routing table entries.

STAT
displays IP routing table statistics, including the total number of entries, the number of active entries, and the number of available entries.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTRTE HELP
  ZTRTE ?
- Each successful ADD, DELETE, or MODIFY results in the filing out of the IP routing table.
- You must remove all references to a given local IP address from the IP routing table before you can remove that local IP address from the TPF system by using the ZTTCP command.
- Do not enter the following IP addresses for the RIP, LIP, NEXTHOP, and NEWLIP parameters:
  - 0.0.0.0
  - 255.255.255.255
  - 127.0.0.0
- Network masks must be continguous ones followed by contiguous zeros at the bit level; for example:
  - 255.255.255.128 is X'FFFFFF80' and is valid.
  - 255.255.255.64 is X'FFFFFF40' and is not allowed.

Examples

The following example adds an IP routing table entry for a specific remote IP address:

```
User:   ZTRTE ADD RIP-9.117.147.165 LIP-9.117.249.70

System: TRTE0001I 09.01.01 IP ROUTING TABLE ENTRY CREATED,   1000 ENTRIES AVAILABLE
```

The following example adds an IP routing table entry for a subnet of remote IP addresses (9.117.147.00–9.117.147.255):

```
User:   ZTRTE ADD RIP-9.117.147.00 LIP-9.117.249.70 NETMASK-255.255.255.0

System: TRTE0001I 09.01.01 IP ROUTING TABLE ENTRY CREATED,   999 ENTRIES AVAILABLE
```

The following example deletes an IP routing table entry for a specific remote IP address:

```
User:   ZTRTE DELETE RIP-9.117.147.165 LIP-9.117.249.70

System: TRTE0002I 09.01.01 IP ROUTING TABLE ENTRY DELETED,   1000 ENTRIES AVAILABLE
```

The following example deletes an IP routing table entry for a subnet of remote IP addresses (9.117.147.00–9.117.147.255):

```
User:   ZTRTE DELETE RIP-9.117.147.00 LIP-9.117.249.70 NETMASK-255.255.255.0

System: TRTE0002I 09.01.01 IP ROUTING TABLE ENTRY DELETED,   1001 ENTRIES AVAILABLE
```

The following example modifies an IP routing table entry for a specific remote IP address, changing its associated local IP address to 9.117.259.72:

```
User:   ZTRTE MODIFY RIP-9.117.147.165 LIP-9.117.249.70 NEWLIP-9.117.259.72

System: TRTE0003I 09.01.01 IP ROUTING TABLE ENTRY MODIFIED,   1001 ENTRIES AVAILABLE
```

The following example displays all IP routing table entries for a specific remote IP address:

```
User:   ZTRTE DISP RIP-9.117.147.165

System: TRTE0004I 09.01.01 IP ROUTING TABLE DISPLAY
        REMOTE           NETMASK        LOCAL            NEXTHOP
        IP                              IP               IP
        --------------- ------------- ----------------- ----------------
        9.117.147.165                   9.117.249.70
        9.117.147.165                   9.117.249.71
        9.117.147.165                                    9.117.247.72
        END OF DISPLAY
```

The following example displays all IP routing table entries for a subnet of remote IP addresses:

```
User:   ZTRTE DISP RIP-9.117.147.0 NETMASK-255.255.255.0

System: TRTE0004I 09.01.01 IP ROUTING TABLE DISPLAY
        REMOTE            NETMASK             LOCAL            NEXTHOP
        IP                                    IP               IP
        --------------- ------------------- ---------------- ------------
        9.117.147.000      255.255.255.000    9.117.249.60
        9.117.147.128      255.255.255.128    9.117.249.50
        9.117.147.165                         9.117.249.70
        9.117.147.165                         9.117.249.71
        9.117.147.165                         9.117.247.72
        END OF DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP support. See *TPF ACF/SNA Network Generation* for more information about the SNAKEY macro.

ZTSTB–Display Tape Status Table Entry

Use this command to display a section of an entry in the tape status table (TSTB).

Requirements and Restrictions

None.

Format

```
                    ┌─1─┐
►►─ZTSTB── ──cuu────┼───┼──────────────────────────────────────────►◄
                    ├─2─┤
                    └─3─┘
```

cuu
is a 3-digit hexadecimal tape device address.

1 displays section 1 of the TSTB entry.

2 displays section 2 of the TSTB entry.

3 displays section 3 of the TSTB entry.

Additional Information

- You can display a TSTB entry for a tape device even if the tape device does not have a tape mounted on it.
- The physical layout of the fields in the TSTB is not necessarily the same as the layout that is displayed by this command.

Examples

Section 1 of the TSTB entry for the 480 tape device is displayed in the following example.

```
User:   ZTSTB 480 1

System: COS30001I 10.10.15 TAPE STATUS TABLE SECTION 1

        ADDRESS   - 001BD05C   HEX LENGTH - 003C   MOD NUM    - 0001
        TAPE NAME - RTA        DEV ADDR   - 0480   TAPE CHAIN - 00
        PRIMARY   - 00         SECONDARY  - 01     TERTIARY   - 20
        SSNAME    - BSS        SSU NAME   - BSS    VOLSER     - A00356
        DEVICE NED- 34803480 00000000 00000000 00000000
                    00000000 00000000 00000000 00000000
```

Section 2 of the TSTB entry for the 480 tape device is displayed in the following example.

```
User:   ZTSTB 480 2

System: COS30003I 11.07.50 TAPE STATUS TABLE SECTION 2

        ADDRESS      - 001BE100   HEX LENGTH - 0100  MOD NUM - 0001
        MOD QUEUE    - 00000000 00847000    BYPASS QUEUE - 00000000 00000000
        QUEUE LENGTH - 00000000
        STATUS FLAGS - 00       ERP FLAGS    - 00    FMT FLAGS    - 10
        CUR FORMAT   - 03       SEIZE FLAG   - 00    SIOSC CC     - 00
        PATH MASK    - 80       FEATURES     - C0    SEIZING PROG - COSA
        BLOCKS       - 00000002             DEVICE TYPE  - 3480
        SENSE LENGTH - 00                   I/O RETRIES  - 0000
        FAILING CCW  - 00000000 00000000    ERROR SCSW   - 00000000 00000000
        DEV DEP DATA - 00000000 00000000 040000E0 0000001D 00000000 00000000
        DOR BLOCK    - 04800500 00808000 00311018 00000000 00847000
        SDR AREA     - xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
        SENSE DATA   - 00000000 00000000 00000000 00000000
                       00000000 00000000 00000000 00000000
        DEFAULT CAT  - xxxx
        USER DATA    - 00000000 00000000
```

Section 3 of the TSTB entry for the 480 tape device is displayed in the following example.

```
User:   ZTSTB 480 3

System: COS40003I 10.13.47 TAPE STATUS TABLE SECTION 3

        ADDRESS  - 001C2100   HEX LENGTH - 0100   MOD NUM - 0001
        TLMR REC - C8C4D9F1 D9E3C14B E3C1D7C5 40404040
                   40404040 40C1F0F0 F3F5F6F0 F0F0F1F0
                   F0F0F1F0 F0F1F2F0 F040F9F4 F0F3F340
                   F0F0F0F0 F7F04040 40404040 D5D6D5C5
                   40404040 40404040 40004040 40404040
        IDAWS    - 00269000 00269800 0026A000 0026A800
                   0026B000 0026B800 0026C000 0026C800
                   0026D000 0026D800 0026E000 0026E800
                   0026F000 0026F800 00270000 00270800
        CUR POS  - 00000000
        TOT LEN  - 00000000
        BUF TIME - 0000
        BUF FLGS - 00
```

Related Information

None.

ZTTCP ACTIVATE–Activate IP Routers

Use this command to do the following:

- Activate a specific Internet Protocol (IP) router or Open Systems Adapter (OSA)-Express connection
- Activate all IP routers associated with a specific local IP address
- Activate all IP routers and OSA-Express connections.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP ACTivate── ──┬─ALL──────────┬──────────────────────────────────────►◄
                       ├─IP-ipaddr────┤
                       ├─SDA-sda──────┤
                       └─OSA-osaname──┘
```

ALL
activates all IP routers and OSA-Express connections that are defined for the TPF system.

IP-*ipaddr*
activates all IP routers that are associated with a specific local IP address, where *ipaddr* is the numeric local IP address.

Note: The only valid IP addresses are those associated with channel data link control (CDLC) IP routers. An IP address can be associated with an OSA-Express connection, but you must specify the OSA parameter to activate a specific OSA-Express connection.

SDA-*sda*
activates a specific IP router, where *sda* is the symbolic device address of the IP router.

OSA-*osaname*
activates a specific OSA-Express connection, where *osaname* is the name of the OSA-Express connection.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTTCP HELP
  ZTTCP ?
- Once you activate an IP router or OSA-Express connection, the TPF system will automatically attempt to reactivate that IP router or OSA-Express connection after an IPL.
- The process of activating an OSA-Express connection takes 16 seconds or more to be completed.

Examples

The following example activates a specific IP router.

```
User:   ZTTCP ACTIVATE SDA-943

System: TTCP0010I 09.17.22 IP ROUTER SDA-0943 ACTIVATION STARTED
        TTCP0203I 09.17.22 IP ROUTER SDA-0943 ACTIVATED
```

The following example activates a specific OSA-Express connection.

```
User:   ZTTCP ACTIVATE OSA-TPFOSA1

System: TTCP0062I 09.17.22 OSA-TPFOSA1 ACTIVATION STARTED
        TTCP0060I 09.17.38 OSA-TPFOSA1 ACTIVATED
```

The following example activates all IP routers associated with a local IP address.

```
User:   ZTTCP ACTIVATE IP-9.117.249.53

System: TTCP0011I 09.19.25 ACTIVATION OF IP ADDRESS 9.117.249.53 STARTED
        TTCP0016I 09.19.25 ACTIVATION OF IP ADDRESS 9.117.249.53 COMPLETED
        TTCP0203I 09.19.25 IP ROUTER SDA-0943 ACTIVATED
```

The following example activates all IP connections that are defined to the TPF system.

```
User:   ZTTCP ACTIVATE ALL

System: TTCP0185I 11.59.55 ACTIVATION OF ALL IP CONNECTIONS STARTED
        TTCP0203I 11.59.55 IP ROUTER SDA-0949 ACTIVATED
        TTCP0060I 12.00.11 OSA-TPFOSA1 ACTIVATED
        TTCP0060I 12.00.11 OSA-TPFOSA2 ACTIVATED
        TTCP0187I 12.00.13 ACTIVATION OF ALL IP CONNECTIONS COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about IP routers, TCP/IP native stack support, and OSA-Express support.

ZTTCP CHANGE–Change Local IP Address Association

Use this command to do the following:

- Change the default local Internet Protocol (IP) address
- Change the local IP address associated with an IP router
- Change which IP routers are allowed to access a local IP address
- Change the maximum packet size (MPS) value for Transmission Control Protocol (TCP) sockets by using a local IP address associated with an IP router.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP CHange── ───────────────────────────────────────────────────────►

►──┬─DEFIP-defipaddr───────────────────────────────────────┬───────────►◄
   ├─IP-ipaddr──────┬───────────────────────┬─┬──────────────┬─┤
   │                └─ ──REstrict-─┬─YES─┬─┘ └─ ──┬──────────┬┘ │
   │                               └─NO──┘         └─MPS-size─┘  │
   └─SDA-sda── ──IP-ipaddr──────────────────────────────────────┘
```

DEFIP-*defipaddr*
: changes the default local IP address, where *defipaddr* is the numeric local IP address.

IP-*ipaddr*
: changes which IP routers are allowed to access a local IP address or changes the local IP address assigned to a specified IP router, where *ipaddr* is the numeric local IP address.

REstrict
: specifies the level of access, where:

  YES
  : indicates that only certain IP routers can access the local IP address.

  NO
  : indicates that all IP routers can access the local IP address.

MPS-*size*
: changes the MPS value for TCP sockets, where *size* is the maximum packet size.

SDA-*sda*
: changes the local IP address assigned to the specified IP router, where *sda* is the symbolic device address of the IP router.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:

ZTTCP HELP
ZTTCP ?

- You must deactivate an IP router by entering the ZTTCP INACTIVATE command before changing the local IP address of the IP router.
- The default IP address can be an IP address associated with IP routers or an Open Systems Adapter (OSA)-Express connection.

Examples

The following example changes the access to a local IP address.

```
User:   ZTTCP CHANGE IP-9.117.249.53 RESTRICT-YES

System: TTCP0003I 09.22.17 LOCAL IP ADDRESS 9.117.249.53 CHANGED
```

The following example changes the local IP address assigned to an IP router.

```
User:   ZTTCP CHANGE SDA-943 IP-9.117.249.50

System: TTCP0004I 09.23.45 IP ROUTER SDA-0943 NOW USING IP-9.117.249.50
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support and OSA-Express support.

ZTTCP CLEAR–Clear TCP/IP Statistics

Use this command to clear TCP/IP statistics.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP CLear── ──STATS──────────────────────────────────────────►◄
```

STATS
clears statistics about resources used by TCP/IP native stack support.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZTTCP HELP
ZTTCP ?

Examples

The following example displays statistics about resources used by TCP/IP native stack support, the request to have those statistics cleared, and the display of the statistics after they have been cleared.

```
User:   ZTTCP DISPLAY STATS

System: TTCP0182I 12.35.23 BEGIN ZTTCP STATS DISPLAY

                             NUMBER   CURRENT MAXIMUM   MAX IN   MAX IN
                             DEFINED   IN USE  IN USE  USE DATE USE TIME
        SOCKET BLOCK ENTRIES         100       12      14    26MAY  12.16.08
        IP MESSAGE TABLE BLOCKS      100       22      34    26MAY  12.16.21

               18824 IP PACKETS SENT
               18383 IP PACKETS RECEIVED
                   0 CHECKSUM ERRORS DETECTED
                   0 IP FRAGMENTS RECEIVED
                   3 TCP MESSAGES RECEIVED OUT OF ORDER
                  44 TCP MESSAGES RETRANSMITTED
                   0 TCP SOCKETS CLEANED UP BECAUSE OF RETRANSMIT TIMEOUTS

        END OF ZTTCP DISPLAY


User:   ZTTCP CLEAR STATS

System: TTCP0183I 12.35.56 COMPLETED, TCP/IP STATS CLEARED


User:   ZTTCP DISPLAY STATS

System: TTCP0182I 12.35.58 BEGIN ZTTCP STATS DISPLAY

                             NUMBER   CURRENT MAXIMUM   MAX IN   MAX IN
                             DEFINED   IN USE  IN USE  USE DATE USE TIME
        SOCKET BLOCK ENTRIES         100       12      12    26MAY  12.35.56
        IP MESSAGE TABLE BLOCKS      100       22      22    26MAY  12.35.56

                   0 IP PACKETS SENT
                   0 IP PACKETS RECEIVED
                   0 CHECKSUM ERRORS DETECTED
                   0 IP FRAGMENTS RECEIVED
                   0 TCP MESSAGES RECEIVED OUT OF ORDER
                   0 TCP MESSAGES RETRANSMITTED
                   0 TCP SOCKETS CLEANED UP BECAUSE OF RETRANSMIT TIMEOUTS

        END OF ZTTCP DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support.

ZTTCP DEFINE–Define IP Router or Local IP Address

Use this command to do the following:

- Define the Internet Protocol (IP) router devices that will connect to the TPF system
- Define the local IP addresses for the TPF system.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
>>──ZTTCP DEFINE── ──────────────────────────────────────────────────────>

               ┌ ──REstrict-NO──┐                   ┌ ──MPS-3840─┐
>──┬─IP-ipaddr─┴────────────────┴──┬────────────┬───┴────────────┴─┬──────><
   │           └ ──REstrict-YES─┘  └ ──DEFAULT─┘    └ ──MPS-size─┘ │
   └─SDA-sda── ──IP-ipaddr──────────────────────────────────────────┘
```

IP-*ipaddr*
: defines a local IP address to the TPF system or assigns the local IP address to the specified IP router, where *ipaddr* is the numeric local IP address.

REstrict
: specifies the level of access, where:

  YES
  : indicates that only certain IP routers can access the local IP address.

  NO
  : indicates that all IP routers can access the local IP address.

DEFAULT
: defines the default local IP address.

MPS-*size*
: defines the maximum packet size (MPS) value for Transmission Control Protocol (TCP) sockets, where *size* is the maximum packet size.

SDA-*sda*
: defines an IP router to the TPF system, where *sda* is the symbolic device address of the IP router.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTTCP HELP
  ZTTCP ?
- You must define a local IP address before you can use the SDA parameter to assign that IP address to IP routers.

Examples

The following example defines a local IP address.

```
User:   ZTTCP DEFINE IP-9.117.249.53

System: TTCP0001I 09.15.16 LOCAL IP ADDRESS 9.117.249.53 DEFINED
```

The following example defines an IP router.

```
User:   ZTTCP DEFINE SDA-943 IP-9.117.249.53

System: TTCP0002I 09.16.27 IP ROUTER SDA-0943 IP-9.117.249.53 DEFINED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support.

ZTTCP DELETE–Delete Local IP Address or IP Router

Use this command to do the following:

- Delete a local Internet Protocol (IP) address from the TPF system
- Delete an IP router from the TPF system.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP DELETE── ──┬─IP-ipaddr─┬─────────────────────────────────────►◄
                     └─SDA-sda───┘
```

IP-*ipaddr*
: deletes the definition of a local IP address, where *ipaddr* is the numeric local IP address.

SDA-*sda*
: deletes an IP router, where *sda* is the symbolic device address of the IP router.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTTCP HELP
  ZTTCP ?
- To delete a local IP address, no IP router can be associated with this local IP address. You must either delete the definitions of those IP routers or change the local IP address assigned to those IP routers.
- You can select a different local IP address to be the default by entering the ZTTCP CHANGE command.
- If more than one local IP address is defined, you cannot delete the default local IP address.
- You must deactivate an IP router by entering the ZTTCP INACTIVATE command before the IP router can be deleted.

Examples

The following example deletes a local IP address.

```
User:   ZTTCP DELETE IP-9.117.249.66

System: TTCP0006I 08.20.36 LOCAL IP ADDRESS 9.117.249.66 DELETED
```

The following example deletes an IP router.

```
User:   ZTTCP DELETE SDA-932

System: TTCP0007I 08.19.33 IP ROUTER SDA-0932 DELETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support.

ZTTCP DISPLAY–Display IP Routers

Use this command to do the following:

- Display information about one or more Internet Protocol (IP) routers, Open Systems Adapter (OSA)-Express connections, or both.
- Display all local IP addresses
- Display resource statistics about TCP/IP native stack support.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP DISplay── ──┬─ALL── ──┬──────────┬──┬──────────────────────────────►◄
                      │         ├─ACTIVE───┤  │
                      │         └─INACTIVE─┘  │
                      ├─IP-ipaddr─────────────┤
                      ├─LOCIPS────────────────┤
                      ├─SDA-sda───────────────┤
                      └─STATS─────────────────┘
```

ALL
: displays all IP routers and OSA-Express connections that are defined in the TPF system. IP router definitions and OSA-Express definitions are displayed in separate tables.

ACTIVE
: displays all IP routers and OSA-Express connections that are active.

INACTIVE
: displays all IP routers and OSA-Express connections that are not active.

IP-*ipaddr*
: displays all IP routers or the OSA-Express connection that is associated with a specific local IP address, where *ipaddr* is the numeric local IP address.

LOCIPS
: displays all local IP addresses defined in the TPF system.

SDA-*sda*
: displays a specific IP router, where *sda* is the symbolic device address of the IP router.

STATS
: displays statistics about resources used by TCP/IP native stack support.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZTTCP HELP
ZTTCP ?

Examples

The following example displays information about all IP routers and OSA-Express connections, where:

SDA
: is the symbolic device address of the IP router.

CURRENT STATUS
: shows whether the current status of the IP router or OSA-Express connection is active or inactive.

DESIRED STATUS
: shows whether the desired status of the IP router or OSA-Express connection is active or inactive.

LOCAL IP ADDR
: is the local IP address associated with the IP router, or the real IP address of the TPF system across this OSA-Express connection.

ROUTER IP ADDR
: is the IP router address. This field is displayed only when the current status of the IP router is active.

TRACE
: shows whether IP trace is active for a given IP router or local IP address, or for a given OSA-Express connection, where:

  ALL
  : indicates that IP trace is active for this IP router or OSA-Express connection because IP trace is active for all resources.

  IP
  : indicates that IP trace is active for this IP router because IP trace is active for the local IP address associated with this IP router or OSA-Express connection.

  NO
  : indicates that IP trace is not active for this IP router or OSA-Express connection.

  OSA
  : indicates that IP trace is active for this OSA-Express connection because IP trace was specifically started for this OSA-Express connection.

  SDA
  : indicates that IP trace is active for this IP router because IP trace was specifically started for this IP router.

OSA NAME
: is the name of the OSA-Express connection.

READ
: is the read SDA associated with the OSA-Express connection. The write SDA associated with the connection is the read SDA plus 1.

DATA
: is the data SDA associated with the OSA-Express connection.

```
User:   ZTTCP DISPLAY ALL

System: TTCP0184I 15.18.32 IP CONNECTIONS DISPLAY

              CURRENT DESIRED
         SDA   STATUS  STATUS   LOCAL IP ADDR    ROUTER IP ADDR  TRACE
         ----  ------  ------  ---------------  ---------------  -----
         0940  ACTIVE  ACTIVE    9.117.249.050    9.117.249.049   IP
         0941  INACT   ACTIVE    9.117.249.050                    IP
         0942  ACTIVE  ACTIVE    9.117.249.050    9.117.249.065   IP
         0930  INACT   ACTIVE    9.117.249.066                    SDA

                   CURRENT DESIRED
         OSA NAME   STATUS  STATUS   LOCAL IP ADDR   TRACE  READ  DATA
         --------   ------  ------  ---------------  -----  ----  ----
         OSA1       ACTIVE  ACTIVE    9.117.250.098   ALL   1A0A  1A09
         OSA1PRIM   INACT   ACTIVE    9.117.230.098   ALL   1A06  1A08
         OSA2BACK   INACT   INACT     9.117.230.045   OSA   0E0A  0E09
       END OF DISPLAY
```

The following example displays information about a specific IP router.

```
User:   ZTTCP DISPLAY SDA-949

System: TTCP0023I 11.40.09 IP CONNECTIONS DISPLAY

              CURRENT DESIRED
         SDA   STATUS  STATUS   LOCAL IP ADDR    ROUTER IP ADDR  TRACE
         ----  ------  ------  ---------------  ---------------  -----
         0949  ACTIVE  ACTIVE    9.117.249.059    9.117.249.049   SDA
       END OF DISPLAY
```

The following example displays information about all IP routers associated with a specific local IP address.

```
User:   ZTTCP DISPLAY IP-9.117.249.59

System: TTCP0023I 11.43.03 IP CONNECTIONS DISPLAY

              CURRENT DESIRED
         SDA   STATUS  STATUS   LOCAL IP ADDR    ROUTER IP ADDR  TRACE
         ----  ------  ------  ---------------  ---------------  -----
         0949  ACTIVE  ACTIVE    9.117.249.059    9.117.249.049   SDA
         0937  INACT   INACT     9.117.249.059                     NO
       END OF DISPLAY
```

The following example displays all local IP addresses that are defined in the TPF system, where:

LOCAL IP ADDR
: is the local IP address.

MPS
: is the maximum packet size.

REST
: indicates the level of access, where:

  YES
  : indicates that only certain IP routers can access the local IP address.

  NO
  : indicates that all IP routers can access the local IP address.

N/A
indicates that restricted access does not apply because the IP address is an OSA IP address.

TRACE
shows whether IP trace is active for a given IP router or local IP address, where:

ALL
indicates that IP trace is active for this local IP address because IP trace is active for all resources.

IP
indicates that IP trace is active for this local IP address because IP trace was specifically started for the local IP address.

NO
indicates that IP trace is not active for this local IP address.

TYPE
shows whether the local IP address is associated with channel data link control (CDLC) routers, is a real IP address of an OSA-Express connection, or is a virtual IP address, where:

CDLC
indicates that the IP address is associated with CDLC IP routers.

OSA
indicates the real IP address of the TPF system across an OSA-Express connection.

VIPA
indicates the IP address is a virtual IP address (VIPA) associated with an OSA-Express connection.

ACT
indicates whether the IP address is active, where:

YES
indicates that the IP address is active.

NO
indicates that the IP address is not active.

OSA NAME
for VIPAs, this is the name of the primary OSA-Express connection; for real OSA IP addresses, this is the name of either the primary or alternate connection.

ALT OSA
is the name of the alternate OSA-Express connection associated with this virtual IP address (VIPA).

ON
indicates the OSA-Express connection to which the VIPA is currently assigned, where:

PRIM
indicates that the VIPA is currently assigned to the primary OSA-Express connection.

ALT
indicates that the VIPA is currently assigned to the alternate OSA-Express connection.

DEFAULT
is the default local IP address.

```
User:   ZTTCP DISPLAY LOCIPS

System: TTCP0021I 10.37.19 LOCAL IP ADDRESS DISPLAY

          LOCAL IP ADDR    MPS  REST  TRACE  TYPE  ACT  OSA NAME  ALT OSA   ON
         ---------------  ----  ----  -----  ----  ---  --------  --------  ----
           9.117.241.139  1492  N/A    ALL   OSA    YES  TPFPRIM1
           9.117.236.132  1492  N/A    ALL   VIPA   YES  TPFPRIM1  TPFBACK1  PRIM
           9.117.236.130  1492  N/A    IP    VIPA   YES  TPFPRIM1  TPFBACK1  PRIM
           9.117.236.131  1492  N/A    IP    VIPA   YES  TPFPRIM1  TPFBACK1  PRIM
           9.117.241.013  1492  N/A    ALL   OSA    YES  TPFBACK1
           9.117.249.052  3840  YES    ALL   CDLC    NO                          DEFAULT
           9.117.249.053  3480   NO    ALL   CDLC   YES
        END OF DISPLAY
```

The following example displays statistics about resources used by TCP/IP native stack support.

```
User:   ZTTCP DISPLAY STATS

System: TTCP0182I 12.35.23 BEGIN ZTTCP STATS DISPLAY

                                 NUMBER   CURRENT MAXIMUM   MAX IN   MAX IN
                                 DEFINED   IN USE  IN USE  USE DATE USE TIME
        SOCKET BLOCK ENTRIES        100       12      14    26MAY  12.16.08
        IP MESSAGE TABLE BLOCKS     100       22      34    26MAY  12.16.21

               18824 IP PACKETS SENT
               18383 IP PACKETS RECEIVED
                   0 CHECKSUM ERRORS DETECTED
                   0 IP FRAGMENTS RECEIVED
                   3 TCP MESSAGES RECEIVED OUT OF ORDER
                  44 TCP MESSAGES RETRANSMITTED
                   0 TCP SOCKETS CLEANED UP BECAUSE OF RETRANSMIT TIMEOUTS

        END OF ZTTCP DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support and OSA-Express support.

ZTTCP INACTIVATE–Deactivate IP Routers

Use this command to do the following:

- Deactivate a specific Internet Protocol (IP) router or Open Systems Adapter (OSA)-Express connection
- Deactivate all IP routers associated with a specific local IP address
- Deactivate all IP routers and OSA-Express connections
- Deactivate all sockets.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP INACTivate── ──┬─ALL─────────┬──────────────────────────►◄
                         ├─IP-ipaddr───┤
                         ├─SDA-sda─────┤
                         ├─OSA-osaname─┤
                         └─SOCKETS─────┘
```

ALL
deactivates all IP routers and OSA-Express connections that are defined to the TPF system.

IP-*ipaddr*
deactivates all IP routers that are associated with a specific local IP address, where *ipaddr* is the numeric local IP address.

Note: The only valid IP addresses are those associated with channel data link control (CDLC) IP routers. An IP address can be associated with an OSA-Express connection, but you must specify the OSA parameter to deactivate a specific OSA-Express connection.

SDA-*sda*
deactivates a specific IP router, where *sda* is the symbolic device address of the IP router.

OSA-*osaname*
deactivates a specific OSA-Express connection, where *osaname* is the name of the OSA-Express connection.

SOCKETS
deactivates all active sockets.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZTTCP HELP
  ZTTCP ?

- Once you deactivate an IP router or OSA-Express connection, the TPF system will not attempt to activate that IP router or OSA-Express connection again after an IPL.

Examples

The following example deactivates an IP router.

```
User:   ZTTCP INACTIVATE SDA-941

System: TTCP0013I 11.22.23 IP ROUTER SDA-0941 DEACTIVATION COMPLETED
```

The following example deactivates a specific OSA-Express connection.

```
User:   ZTTCP INACTIVATE OSA-TPFOSA1

System: TTCP0061I 11.22.23 OSA-TPFOSA1 HAS BEEN DEACTIVATED
```

The following example deactivates all IP routers associated with a local IP address.

```
User:   ZTTCP INACTIVATE IP-9.117.249.52

System: TTCP0014I 11.24.43 INACTIVATION OF IP ADDRESS 9.117.249.52 STARTED
        TTCP0018I 11.24.45 INACTIVATION OF IP ADDRESS 9.117.249.52 COMPLETED
```

The following example deactivates all IP connections.

```
User:   ZTTCP INACTIVATE ALL

System: TTCP0186I 12.00.40 INACTIVATION OF ALL IP CONNECTIONS STARTED
        TTCP0188I 12.00.40 INACTIVATION OF ALL IP CONNECTIONS COMPLETED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support and OSA-Express support.

ZTTCP TRACE–IP Trace Facility

Use this command to do the following:

- Start or stop the Internet Protocol (IP) trace for one or more IP routers
- Start or stop the writing of the IP trace data to tape
- Start or stop the tracing of Routing Information Protocol (RIP) messages
- Change the amount of data in each message that will be traced.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZTTCP TRace── ──┬─┬─START─┬─ ─┬─ALL──────┬─┬──────────────────────►◄
                    │ └─STOP──┘   ├─IP-ipaddr─┤ │
                    │             ├─SDA-sda───┤ │
                    │             └─TAPE──────┘ │
                    │  ┌ ──SIZE-8────┐          │
                    ├──┴ ──SIZE-size─┴──────────┤
                    │  ┌ ──NORIP─┐              │
                    └──┴ ──RIP───┴──────────────┘
```

START
: starts the IP trace for the specified resources.

STOP
: stops the IP trace for the specified resources.

ALL
: starts or stops the IP trace for all IP routers.

IP-*ipaddr*
: starts or stops the IP trace for all IP routers that are associated with the specified local IP address, where *ipaddr* is the numeric local IP address.

SDA-*sda*
: starts or stops the IP trace for a specific IP router, where *sda* is the symbolic device address of the IP router.

TAPE
: starts or stops writing the IP trace data to tape.

SIZE-*size*
: changes the amount of data in each message that is traced, where *size* is the amount of data in each message to trace. The value can be from 8 to 3840 bytes.

NORIP
: specifies that Routing Information Protocol (RIP) messages are not placed in the trace table.

RIP
: specifies that RIP messages are placed in the trace table.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZTTCP HELP
ZTTCP ?

Examples

The following example activates the IP trace for an IP router.

```
User:   ZTTCP TRACE START SDA-944

System: TTCP0031I 11.29.47 IP TRACE FOR IP ROUTER SDA-0944 STARTED
```

The following example stops the IP trace for all IP routers associated with a local IP address.

```
User:   ZTTCP TRACE STOP IP-9.117.249.54

System: TTCP0040I 11.30.32 IP TRACE FOR LOCAL IP ADDRESS 9.117.249.54 STOPPED
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP native stack support.

ZTVAR–Configure Tape Devices

Use this command to add tape devices to the current tape configuration or to delete tape devices from the current tape configuration.

Requirements and Restrictions

- You can enter this command only from the basic subsystem (BSS).
- Tape hardware restart must be completed.
- You cannot configure a tape device if it is currently being used.

Format

```
►►──ZTVAR── ──┬─Add────┬──cu──┬─────────────────────┬──────────────────►◄
              └─Delete─┘      └─address──┬──────────┬┘
                                         └─-address─┘
```

Add
adds a tape device to the current tape configuration.

Delete
deletes a tape device from the current tape configuration.

cu is the 2-digit hexadecimal address of the control unit for which you are defining the tape device.

address
is the 1-digit hexadecimal address of the tape device. If you omit this parameter, all the tape devices on the control unit (0–F) are configured without regard to other factors.

Note: Specify a second address to add or delete a range of tape devices.

Additional Information

None.

Examples

The 482 and 483 tape devices are configured in the following example.

```
User:   ZTVAR A 482-3

System: COTU0101I 12.55.36 TVAR BSS    PROT ENTRY ACQUIRED FOR DEVICE 482
                           BY THIS PROCESSOR
        COTU0101I 12.55.36 TVAR BSS    PROT ENTRY  ADDED   FOR DEVICE 483
                           BY THIS PROCESSOR
        COTE0002I 12.55.36 TVAR      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         481      RTA    BSS      AO    00 01 00  A00050  38K         1 YES
         480      JEP    HPN      AI    01 03 00  A00049  38K         0 YES
         482     AVAIL
         483     AVAIL
        END OF DISPLAY
```

The 484 tape device is deleted in the following example.

```
User:   ZTVAR D 484

System: COTU0101I 09.47.05 TVAR BSS    PROT ENTRY RELEASED FOR DEVICE 484
                           BY THIS PROCESSOR
        COTE0002I 12.55.36 TVAR      - TAPE STATUS

        ADDRESS   NAME   SSU    STATUS   TPIND    VOLSER  FORMAT  #BLOCKS LDR

         480      RTA    BSS      AO    00 01 30  A00034  38K         7 YES
         481     AVAIL
         482      RTC    BSS      AO    00 01 00  A00145  38K      4391 YES
         483     AVAIL
        END OF DISPLAY
```

Related Information

None.

ZTWTM–Write Tapemarks

Use this command to write 2 tapemarks to a tape to allow that tape to be processed for input after a hardware or software error occurs.

Requirements and Restrictions

- You can enter this command only from the basic subsystem (BSS).
- Use this command only when you know that a tape was not properly closed. Normally, this condition occurs only during a system failure or when the tape control program cannot successfully write trailer labels to an output tape, in which case you are notified by the TPF system.
- This function does not check for unexpired tapes or move the tape in any direction before writing the tapemarks.

Format

```
>>──ZTWTM── ──cuu──┬────────┬────────────────────────────────────────────><
                   └─ ──BP──┘
```

cuu
: is a 3-digit hexadecimal tape device address.

BP
: allows you to write tapemarks to an open tape.

Additional Information

This command uses 10 times the normal number of retries while attempting to write a tapemark. Therefore, it sometimes succeeds even if the tape labeling routines (for example) failed to write trailer labels.

Examples

Tapemarks are written to the tape on the 482 tape device in the following example.

```
User:   ZTWTM 482

System: COTZ0044I 12.11.06 TWTM BSS    COMPLETE
```

Related Information

None.

ZVFAC–Enable or Disable Delay File Support

Use this command to enable or disable delay file support for virtual file access (VFA).

Requirements and Restrictions

You can enter this command in any machine state. However, it does not take effect until the system reaches NORM state because delay file support is disabled if the TPF system is not in NORM state.

Format

```
►►──ZVFAC── ──┬─Enable──┬──────────────────────────────────────────────►◄
              └─DISAble─┘
```

Enable
enables delay file support for VFA.

DISAble
disables delay file support for VFA.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZVFAC HELP
ZVFAC ?

Examples

Delay file support is enabled in the following example.

```
User:   ZVFAC ENABLE

System: VFAC0015I 07.30.32 DELAYED FILES ENABLED
```

Delay file support is disabled in the following example.

```
User:   ZVFAC DISABLE

System: VFAC0016I 07.30.32 DELAYED FILES DISABLED BY OPERATOR
```

Related Information

See *TPF Database Reference* for more information about VFA and delay file support.

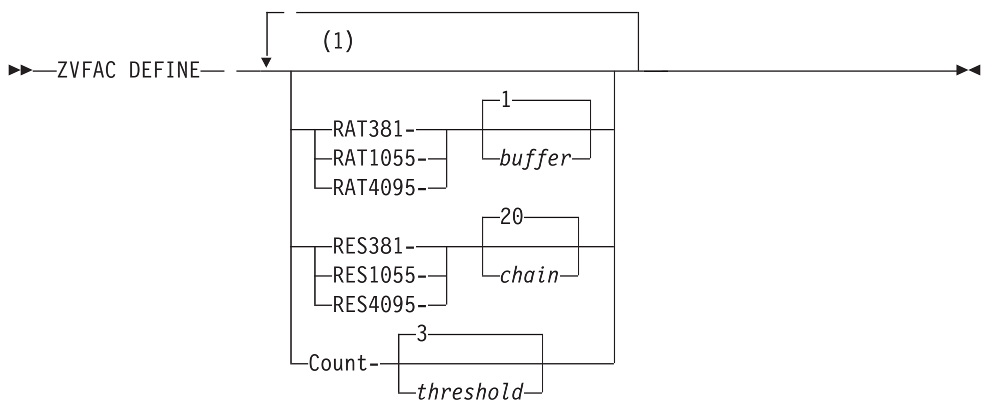
ZVFAC DEFINE–Define VFA Resources

Use this command to change buffer ratios, reserve chain sizes, and define the buffer reuse threshold value for virtual file access (VFA).

Requirements and Restrictions

You can enter this command only for the basic subsystem (BSS).

Format



Notes:

1 You can also separate the parameters with the , character.

RAT381
defines the 381 VFA buffer ratio.

RAT1055
defines the 1055 VFA buffer ratio.

RAT4095
defines the 4095 VFA buffer ratio.

buffer
is a buffer ratio from 1–999.

RES381
defines the reserve chain size for the 381 VFA buffer.

RES1055
defines the reserve chain size for the 1055 VFA buffer.

RES4095
defines the reserve chain size for the 4095 VFA buffer.

chain
is a reserve chain size percentage from 1–99. The reserve chain destages the VFA buffers that are delay file pending before the buffers are needed by VFA for new records.

The recommended values are:
- 5 percent for a production system

- 10 percent for a test system.

Count-*threshold*
defines the buffer reuse threshold value, where *threshold* is a value from 1–999.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZVFAC HELP
  ZVFAC ?
- Specify only the parameters that you want to change. If you do not specify a parameter, it retains its current value.
- The value specified for the COUNT parameter takes effect immediately. All other values do not take effect until the TPF system rebuilds the VFA area.
- You can abbreviate the parameters to the first 4 characters. For example, you can enter RAT1055 as RAT1.
- The buffer values are stored in keypoint record A (CTKA).

Examples

The buffer ratios, reserve chain sizes, and the reuse threshold value are changed in the following example.

```
User:   ZVFAC DEF RAT1-10 RAT3-5 RAT4-3 RES1-5 RES3-10 RES4-15 C-36

System: VFAC0006I 13.12.15  VFA KEYPOINT VALUES ARE:
             RAT1055- 10, RAT381-  5, RAT4095-  3
             RES1055-  5, RES381- 10, RES4095- 15
             COUNT- 36
                         CURRENT VFA VALUES ARE:
             RAT1055-  1, RAT381-  1, RAT4096-  1
             RES1055- 20, RES381- 20, RES4096- 20
             COUNT- 36
```

Related Information

See *TPF Database Reference* for more information about VFA, buffer ratios, reserve chain sizes, and the reuse threshold value.

ZVFAC DISPLAY–Display VFA Resources

Use this command to display the following information about virtual file access (VFA):

- Current status of VFA
- Buffer ratios and reserve chain values
- Buffer status.

Requirements and Restrictions

None.

Format

```
                      ┌─Status───┐
►►──ZVFAC DISPlay── ──┼──────────┼──────────────────────────────────►◄
                      ├─Resource─┤
                      ├─Buffer───┤
                      └─All──────┘
```

Status
displays the current status of VFA.

Resource
displays the buffer ratios and reserve chain values.

Note: You can specify this parameter only for the basic subsystem (BSS).

Buffer
displays the buffer status.

All
displays status, resource, and buffer information.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZVFAC HELP
ZVFAC ?

Examples

The current status of VFA is displayed in the following example.

```
User:   ZVFAC DISP STA

System: VFAC0017I 07.59.27 DELAYED FILES DISABLED BY SYSTEM
```

The buffer ratios and reserve chain values are displayed in the following example.

```
User:   ZVFAC DISP RES

System: VFAC0006I 07.59.27 VFA KEYPOINT VALUES ARE:
             RAT1055- 10, RAT381-  5, RAT4095-  3
             RES1055-  5, RES381- 10, RES4095- 15
             COUNT- 36
                         CURRENT VFA VALUES ARE:
             RAT1055-  1, RAT381-  1, RAT4095-  1
             RES1055- 20, RES381- 20, RES4095- 20
             COUNT- 36
```

The buffer status is displayed in the following example.

```
User:   ZVFAC DISP BUF

System: VFAC0086I 17.08.51 VFA BUFFER STATUS
             BUFFER TYPE                4K    LARGE    SMALL
             NUMBER OF BUFFERS        4453     4557     4565
             NUMBER ON RESERVE         219      298      242
             NUMBER ON AGING          4334     4259     4323
             NUMBER IN IO WAIT           0        0        0
             NUMBER IN COMMIT            0       48        8
             PROGRAM RECORDS          2250        0        0
             DELAYED-FILE PENDING       13       16       10
             VFA_S LOCK BUFFERS         24       11       26
             VFA_X LOCK BUFFERS         13       16       10
             CACHED RHT LOCKS            9        9        4
```

All the status, resource, and buffer information is displayed in the following example.

```
User:   ZVFAC DISP ALL

System: ZVFAC DISP ALL
        VFAC0017I 07.59.27 DELAYED FILES DISABLED BY SYSTEM
        VFAC0006I 07.59.27 VFA KEYPOINT VALUES ARE:
             RAT1055- 10, RAT381-  5, RAT4095-  3
             RES1055-  5, RES381- 10, RES4095- 15
             COUNT- 36
                         CURRENT VFA VALUES ARE:
             RAT1055-  1, RAT381-  1, RAT4095-  1
             RES1055- 20, RES381- 20, RES4095- 20
             COUNT- 36
        VFAC0086I 07.59.27 VFA BUFFER STATUS
             BUFFER TYPE                4K    LARGE    SMALL
             NUMBER OF BUFFERS        1581     1582     1589
             NUMBER ON RESERVE         370      473      324
             NUMBER ON AGING          1211     1109     1265
             NUMBER IN IO WAIT           0        0        0
             NUMBER IN COMMIT            0        0        0
             PROGRAM RECORDS           360        0        0
             DELAYED-FILE PENDING        0        0        0
             VFA-S LOCK BUFFERS          0        0        0
             VFA-X LOCK BUFFERS         19        0        0
             CACHED RHT LOCKS            5        0        0
```

Related Information

See *TPF Database Reference* for more information about VFA, buffer ratios, reserve chain sizes, and the reuse threshold value.

ZVFAC FLUSH–Flush VFA

Use this command to remove a file address or record ID from virtual file access (VFA). The record is written to DASD if it was marked for delay filing.

Requirements and Restrictions

You cannot remove a program at a specified file address from VFA if that program is in use (locked in core).

Format

```
►►──ZVFAC FLUsh── ──┬─FA── ──fileaddr─┬──────────────────────────────────►◄
                    └─RECid── ──id────┘
```

FA *fileaddr*
: removes the specified file address from VFA, where *fileaddr* is the 8- or 16-digit hexadecimal file address.

RECid *id*
: removes the specified record ID from VFA, where *id* is the 4-digit hexadecimal record ID.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZVFAC HELP
  ZVFAC ?
- Commit scope records that reside in VFA buffers are not VFA records and are, therefore, not affected by this command.

Examples

File address 8850000F is removed from VFA in the following example.

```
User:   ZVFAC FLU FA 8850000F

System: VFAC0112I 22.30.30 FLUSH REQUEST COMPLETED
                           FILE ADDRESS 000000008850000F
```

Record ID 00FF is removed from VFA in the following example.

```
User:   ZVFAC FLUSH REC 00FF

System: VFAC0011I 23.30.30 FLUSH REQUEST COMPLETED FOR RECORD ID 00FF,
        NUMBER RECORDS FLUSHED 293
```

Related Information

See *TPF Database Reference* for more information about VFA and delay filing.

ZVFAC INDICATE–Indicate VFA Measurements

Use this command to display information about the efficiency of virtual file access (VFA).

Requirements and Restrictions

None.

Format

```
►►──ZVFAC Indicate──────────────────────────────────────────────────►◄
```

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZVFAC HELP
ZVFAC ?

Examples

Information about the efficiency of VFA is displayed in the following example.

```
User:   ZVFAC IND

System: VFAC0024I 17.26.20 VFA EFFICIENCY STATISTICS

          FAST RSHT'S IN USE (PERCENT)       3
          AVERAGE ALIAS CHAIN LENGTH      1.04
          LONGEST ALIAS CHAIN                2

          MASTER RSHT'S IN USE (PERCENT)     3
          AVERAGE ALIAS CHAIN LENGTH      1.02
          LONGEST ALIAS CHAIN                2

          PROG READS      0 PER SEC.
          DATA READS    913
          FIND I/O       49
          FILE CAND     913
          FILE IMMD     913
          FORCE FILE      0
          FILE NCAND      1
          REJECTED        0
          381  USAGE      2
          1055 USAGE      3
          4K   USAGE     44


          VFA-S HITS           0
          VFA-X HITS        1786
          VFA LOCK MISSES     43
          VFA CONTENTIONS     21
          RHT CONTENTIONS     21
          CACHED RHT HITS    882
          VFA LOCK AGEOUTS     0

          INTERVAL      147 SECS.+
```

Related Information

See *TPF Database Reference* for more information about VFA.

ZVFAC LOCATE–Locate VFA Residents

Use this command to locate virtual file access (VFA) residents and display the following information:
- Buffer control area (BCA) core address
- VFA buffer core address
- Buffer type
- Filing status
- Program record, if the resident is a file resident program.

Requirements and Restrictions

None.

Format

```
>>──ZVFAC LOCate── ──┬─Program── ──progname──────────────────────┬──────────────><
                     │                          ┌─.0──────┐      │
                     ├──┬─FACS── ──symbolic─┬───┤         ├──────┤
                     │  └─FACE── ──rectype──┘   └─.ordnum─┘      │
                     ├─FADdress── ──fileaddr─────────────────────┤
                     └─MCHR── ──mmmmcccchhhhrr───────────────────┘
```

Program *progname*
: locates the most recently activated version of a program, where *progname* is the 4-character alphanumeric name of the program.

FACS *symbolic*
: locates a fixed file record, where *symbolic* is the 8-character alphanumeric symbolic file address compute (FACE) program ID.

FACE *rectype*
: locates a fixed file record, where *rectype* is the 4-digit hexadecimal FACE ID.

ordnum
: is a 1- to 16-digit hexadecimal ordinal number.

FADdress *fileaddr*
: locates a FARF file address, where *fileaddr* is the 8- or 16-digit hexadecimal file address.

MCHR *mmmmcccchhhhrr*
: locates an MCHR file address, where

  mmmm
  : is the 4-digit hexadecimal symbolic module number

  cccc
  : is the 4-digit hexadecimal cylinder number

  hhhh
  : is the 4-digit hexadecimal head number

  rr
  : is the 2-digit hexadecimal record number.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZVFAC HELP
  ZVFAC ?
- Commit scope records that reside in VFA buffers are not VFA records and are, therefore, not affected by this command.

Examples

The CVAU program is located in the following example.

```
User:   ZVFAC LOC PROG CVAU

System: VFAC0111I 07.59.27 RECORD STATUS IN VFA
        BCA ADDRESS     - 0097F8C8
        VFA BUFFER ADDRESS - 00A4E000
        VFA FILE ADDRESS - 000000004408371F
        PROGRAM RECORD
        4K VFA BUFFER
        BCA IS IN FAST LOOKUP TABLE
        END OF DISPLAY
```

The fixed file record with FACE ID 0A3 and ordinal number 1 is located in the following example.

```
User:   ZVFAC LOC FACE 0A3.1

System: VFAC0111I 10:11:31 RECORD STATUS IN VFA
        BCA ADDRESS -       00FE1078
        VFA BUFFER ADDRESS - 00FF8000
        VFA FILE ADDRESS - 000000005509482F
        1055 VFA BUFFER
        BCA IS IN FAST LOOKUP TABLE
        DELAYED FILE PENDING
```

The fixed file record with FACE ID 0D8 and ordinal number 0 for a VFA candidate is located in the following example.

```
User:   ZVFAC LOC FACE 0D8.0

System: VFAC0111I 12.27.48 RECORD STATUS IN VFA
        BCA ADDRESS -       01A82C30
        VFA BUFFER ADDRESS - 02493000
        VFA FILE ADDRESS - 00000000CC03A800
        4K VFA BUFFER
        BCA IS IN FAST LOOKUP TABLE
        IMMEDIATE FILE
        END OF DISPLAY
```

The fixed file record with FACE ID 0DA and ordinal number 1 for a VFA synchronization candidate is located in the following example.

ZVFAC LOCATE

```
User:   ZVFAC LOC FACE 0DA.1

System: VFAC0111I 12.27.48 RECORD STATUS IN VFA
        BCA ADDRESS -        01A81BB0
        VFA BUFFER ADDRESS - 0247D000
        VFA FILE ADDRESS - 00000000CC03A801
        4K VFA BUFFER
        BCA IS IN FAST LOOKUP TABLE
        VFA-X LOCK HELD
        RHT LOCK CACHED IN BCA
        IMMEDIATE FILE
        END OF DISPLAY
```

Related Information

See *TPF Database Reference* for more information about VFA.

ZVFAC PURGE–Purge VFA

Use this command to remove a file address or record ID from virtual file access (VFA). The record is ***not*** written to DASD if it was marked for delay filing.

Requirements and Restrictions

None.

Format

```
►►──ZVFAC PURGE── ──┬─FA── ──fileaddr─┬────────────────────────────────────►◄
                    └─RECid── ──id────┘
```

FA *fileaddr*
: removes the specified file address from VFA, where *fileaddr* is the 8- or 16-digit hexadecimal file address.

RECid *id*
: removes the specified record ID from VFA, where *id* is the 4-digit hexadecimal record ID.

Additional Information

- Online help information is available for this command. To display the help information, enter one of the following:
  ZVFAC HELP
  ZVFAC ?
- Commit scope records that reside in VFA buffers are not VFA records and are, therefore, not affected by this command.

Examples

File address 000000008850000F is removed from VFA in the following example.

```
User:   ZVFAC PURGE FA 000000008850000F

System: VFAC0113I 22.33.30 PURGE REQUEST COMPLETED
                           FILE ADDRESS 000000008850000F
```

Record ID 00FF is removed from VFA in the following example.

```
User:   ZVFAC PURGE REC 00FF

System: VFAC0013I 20.30.31  PURGE REQUEST COMPLETED FOR RECORD ID 00FF,
        NUMBER RECORDS PURGED 293
```

Related Information

See *TPF Database Reference* for more information about VFA and delay filing.

ZVIPA–Manage a Virtual IP Address

Use this command to do the following:

- Display TCP/IP traffic information about individual central processing units (CPUs), virtual IP addresses (VIPAs), Open Systems Adapter (OSA) IP addresses, and other statistical information
- Move a VIPA from one processor to another processor in the same loosely coupled complex
- Summarize system workload levels and VIPA statistics.

Requirements and Restrictions

- You can enter this command only in 1052 state or higher.
- You can enter this command only from the basic subsystem (BSS).

Format

```
►►──ZVIPA── ──┬─DISplay── ──┬─IP-ipaddr──────────────────┬─┬──────────────────────►◄
              │             ├─CPU-cpuid─┬─────────────┬──┤ │
              │             │           ├─ACTIVE───┤  │  │ │
              │             │           └─INACTive─┘  │  │ │
              │             ├─TYPE-type─┬──────────────┬─┤ │
              │             │           └─ CPU-cpuid─┘   │ │
              │             └─ALL────────────────────────┘ │
              ├─MOVE── ──VIPA-vipaddr── ──CPU-cpuid─────────┤
              └─SUMmary─────────────────────────────────────┘
```

DISplay
: displays statistical information for a particular Internet Protocol (IP) address, a set of IP addresses on a CPU, or specific types of IP addresses on a particular CPU or complex.

IP-*ipaddr*
: specifies the OSA IP address, where *ipaddr* is a numeric IP address.

CPU-*cpuid*
: specifies the processor ID of a TPF processor, where *cpuid* is a valid TPF processor ID defined in the complex.

ACTIVE
: displays only active IP addresses.

INACTive
: displays only inactive IP addresses.

TYPE-*type*
: displays the type of IP addresses, where *type* is one of the following:

  MOVABLE
  : displays only VIPAs that are defined as movable.

  STATIC
  : displays only VIPAs that are defined as static.

  REAL
  : displays only real Open Systems Adapter (OSA) IP addresses.

VIPA
displays all VIPAs.

ALL
displays all OSA IP addresses in the TPF complex.

MOVE
moves a VIPA from one processor to another in the same loosely coupled complex.

VIPA-*vipaddr*
moves the VIPA, where *vipaddr* is a numeric IP address.

SUMmary
displays system and VIPA statistics.

Additional Information

Online help information is available for this command. To display the help information, enter one of the following:

ZVIPA HELP
ZVIPA ?

Examples

The following example displays system and OSA IP address statistics, where:

CPU
is the processor.

PACKETS/SEC
indicates the number of messages sent and received per second as determined by the traffic for the previous minute.

CPU UTIL
shows the immediate CPU utilization averaged over the I-streams for the designated processor.

```
User:   ZVIPA SUM

System: VIPA0003I 15.18.32 OSA IP ADDRESS SUMMARY DISPLAY BEGINS

          CPU PACKETS/SEC CPU UTIL
          --- ----------- --------
           A         2300   76.9
           B          290   12.1
           C          650    1.2
           E          122   21.2

        END OF DISPLAY
```

The following example displays information about OSA IP addresses defined on CPU B, where:

CPU
is the processor that currently owns the IP address.

IP is the IP address.

TYPE
is the type of IP address, where *type* is one of the following:

MOVABLE
specifies an IP address that is defined as a movable VIPA.

STATIC
: specifies an IP address that is defined as a static VIPA.

REAL
: specifies a real OSA IP address.

MOVING TO CPU
: is the CPU to which the movable VIPA is in the process of moving.

ACTIVE
: indicates whether the IP address is active.

PACKETS/SEC
: indicates the number of messages sent and received per second as determined by the traffic for the previous minute.

```
User:   ZVIPA DISPLAY TYPE-VIPA CPU-B

System: VIPA0002I 15.18.34 OSA IP ADDRESS DISPLAY BEGINS
                                               MOVING
          CPU       IP             TYPE        TO CPU   ACTIVE  PACKETS/SEC
          ---  --------------     ---------    ------   ------- -----------
           B     9.117.249.68     MOVABLE                YES            234
           B     9.117.249.69     REAL                   YES             42
           B     9.117.249.70     MOVABLE                YES            534
           B     9.117.249.71     STATIC                 YES             75
           B     8.100.200.22     MOVABLE                 NO              0
           B     8.100.200.24     STATIC                  NO              0

        END OF DISPLAY
```

The following example displays information about an OSA IP address and which CPU the IP address is currently defined on, where:

CPU
: is the processor that currently owns the IP address.

IP
: is the IP address.

TYPE
: is the type of IP address, where *type* is one of the following:

MOVABLE
: specifies an IP address that is defined as a movable VIPA.

STATIC
: specifies an IP address that is defined as a static VIPA.

REAL
: specifies a real OSA IP address.

MOVING TO CPU
: is the CPU to which the movable VIPA is in the process of moving.

ACTIVE
: indicates if the IP address is active.

PACKETS/SEC
: indicates the number of messages sent and received per second as determined by the traffic for the previous minute.

DEFINED TO CPU
: is the processor or processors to which the movable VIPA is defined.

```
User:   ZVIPA DISPLAY IP-9.117.249.68

System: VIPA0010I 15.18.34 OSA IP ADDRESS DISPLAY BEGINS
                                           MOVING
          CPU       IP             TYPE       TO CPU   ACTIVE  PACKETS/SEC
          ---  --------------     ---------    ------   ------- -----------
           B     9.117.249.68     MOVABLE               YES            234

           DEFINED TO CPU - B C

        END OF DISPLAY
```

The following example moves a VIPA from one processor to another in the same loosely coupled complex.

```
User:   ZVIPA MOVE VIPA-9.117.249.70 CPU-C

System: VIPA0004I 19:39:58 VIPA-9.117.249.70 MOVING FROM CPU B TO CPU C
        VIPA0001I 19:39:59 VIPA-9.117.249.70 MOVED FROM CPU B TO CPU C

        END OF DISPLAY
```

Related Information

See *TPF Transmission Control Protocol/Internet Protocol* for more information about OSA-Express support.

Part 2. System Utilities Operations

Diagnostic Output Formatter

The offline diagnostic output formatter (DOF) formats and prints data (postprocesses) on the real-time log tape.

Input Files

Input to the DOF is the real-time log tape (RTL/RTA) that is created during normal real-time operations or while running the program test vehicle (PTV).

Job Control Language

The job control language (JCL) required for DOF includes the following statements:

```
//STEPNAME  EXEC  PGM=PPCP,REGION=512K
//STEPLIB   DD    DSN=ACP.LINK.RELvv,DISP=SHR
//PRDD      DD    SYSOUT=A
//SYSOUT    DD    SYSOUT=A
//DDSCTH    DD    DSN=&&SCRTH,DISP=(NEW,DELETE),UNIT=(SYSDA,,DEFER),
//                SPACE=(TRK,(50,200))
//SYS000    DD    DSN=RTn.TAPE,DISP=OLD,UNIT=TAPE,
//                VOL=SER=xxxxxx,DCB=OPTCD=B
```

Processing Multivolume Tapes

If you are processing only one tape, skip this box. If you are processing more than one tape, code the following JCL statements:

```
//SYS000    DD    DSN=RTn.TAPE,UNIT=TAPE,LABEL=(,SL),DISP=(SHR,PASS),
//                VOL=SER=(xxxxxx,yyyyyy),DCB=OPTCD=B
```

```
//SORTIN    DD    DSN=&&SRTIN,DISP=(NEW,DELETE),
//                DCB=(RECFM=VB,LRECL=3700,BLKSIZE=3704),
//                UNIT=(SYSDA,,DEFER),SPACE=(TRK,(50,200))
//SORTOUT   DD    DSN=&&SRTOUT,DISP=(NEW,DELETE),
//                DCB=(RECFM=VB,LRECL=3700,BLKSIZE=3704),
//                UNIT=(SYSDA,,DEFER),SPACE=(TRK,(50,200))
//SORTLIB   DD    DSN=SYS1.SORTLIB,DISP=SHR
//SORTWK01  DD    UNIT=(SYSDA,,DEFER),SPACE=(TRK,(150),,CONTIG)
//SORTWK02  DD    UNIT=(SYSDA,,DEFER),SPACE=(TRK,(150),,CONTIG)
//SORTWK03  DD    UNIT=(SYSDA,,DEFER),SPACE=(TRK,(150),,CONTIG)
//SYSUDUMP  DD    SYSOUT=A
/*
//
```

EXEC Statement

The size of the REGION parameter in the EXEC statement is based on the largest message that will be processed. The largest message size is derived from the SIP SYGLB/SYSET value for &SAM3270. The minimum tested values are 512K when the SM (sort/merge) option is specified for DOF and 250K when the SM option is not specified. These values are used with the assumption that the largest message is approximately 3600 bytes plus control characters. See “Third Message for DOF Options (TV Mode)” on page 1442 for more information about the SM option.

You can decrease the size of the REGION parameter when DOF is generated with less than full capability.

You can also use the EXEC statement to specify parameters for the DOF options instead of typing them in response to console messages. See “Specifying DOF Options in the JCL EXEC Statement” on page 1447 for more information.

STEPLIB Data Definition

The STEPLIB data definition refers to the load module library, and *vv* identifies the version number of the TPF release. It may be necessary to concatenate a load module library from an earlier release to access the global label table for the global area labels.

SYS000 Data Definition

The SYS000 data definition refers to the input RTA/RTL tape generated by any TPF release since ACP5. The *n* is an A or an L to define whether an RTA or an RTL tape is being used.

You must change *xxxxxx* in the VOL=SER= parameter to represent the volume serial number (VSN) of the tape being that is processed. If the data for a single system error dump is contained on more than one volume, you have two choices for coding the JCL:

1. Concatenate the data sets using multiple data definitions (DD) statements. Code the VOL=SER= parameter on the first DD statement as VOL=SER=*xxxxxx*, where *xxxxxx* represents the VSN of the tape that contains the first data set. Code the VOL=SER= parameter on the second data definition statement as VOL=SER=*yyyyyy*, where *yyyyyy* represents the VSN of the tape that contains the second data set. Code the DISP= parameter as (SHR,PASS), where SHR indicates that the data existed before this job step and can be shared. PASS indicates that the data set will be passed to a subsequent job step in the same job.
2. Code the VOL=SER= parameter as VOL=SER=(*xxxxxx*,*yyyyyy*), where *xxxxxx* and *yyyyyy* represent the VSNs of the tapes that contain the data. These VSNs must be listed in the order in which they were created. Code DCB=OPTCD=B to disregard the end-of-file condition. OPTCD=B prevents a premature end-of-file indication for multivolume input data sets.

SORT Data Definitions

Data definitions that start with SORT are used with the SM (sort/merge) option. These data definitions are ignored if you do not specify the SM option or if the TPF system does not support PTV.

If you specify the NOPR option, you may want to modify the SORTOUT data definition to a KEPT data set, either tape or DASD.

See “Third Message for DOF Options (TV Mode)” on page 1442 for more information about the SM option and the NOPR option.

You must also code the DCB parameter for the SORTIN and SORTOUT data definitions. The values you assign to the LRECL and BLKSIZE parameters vary according to the largest message plus any control characters. When you assemble the BMP0 segment, an MNOTE statement is generated after calculating the LRECL and BLKSIZE parameters. The values previously shown represent a maximum TPF message size of 3600 bytes plus any control characters. This value is calculated from the SIP SYGLB/SYSET value for &SAM3270.

Using a Cataloged Procedure

The JCL can be placed in the MVS SYS1.PROCLIB without the /* card and, thereby, eliminate the use of control cards. If this is done, the following run deck is the only run deck required:

```
//          JOB
//          EXEC  Procedure name
/*
```

Link-Edit Procedures

The following statements are required for the link edit:

```
//LKED      EXEC   PGM=IEWLF440,REGION=88K,PARM='LET,LIST,XREF,DCBS,
//                 SIZE=(88K,6K),NCAL,OVLY'
//SYSPRINT DD      SYSOUT=A,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=605)
//SYSLMOD  DD      DSN=ACP.LINK.RELvv,DISP=OLD,DCB=BLKSIZE=1200
//OBJLIB   DD      DSN=ACP.OBJ.RELvv,DISP=SHR
//SYSUT1   DD      DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(7294,(100,100))
//SYSLIN   DD      *
           ENTRY   PPCP
          INCLUDE  OBJLIB(PPCPnn)
          INCLUDE  OBJLIB(CCMCDCnn)
          OVERLAY  ONE
          INCLUDE  OBJLIB(STPPnn)
          OVERLAY  ONE
          INCLUDE  OBJLIB(BMP0nn)
          OVERLAY  TWO
          INCLUDE  OBJLIB(BMP1nn)
          OVERLAY  TWO
          INCLUDE  OBJLIB(BMP2nn)
          OVERLAY  TWO
          INCLUDE  OBJLIB(BMP3nn)
          OVERLAY  TWO
          INCLUDE  OBJLIB(BMP4nn)
          OVERLAY  TWO
          INCLUDE  OBJLIB(BMP5nn)
          OVERLAY  THREE
          INCLUDE  OBJLIB(BMP6nn)
          OVERLAY  THREE
          INCLUDE  OBJLIB(BMP7nn)
          NAME     PPCPnn(R)
/*
```

The global label table must be included in the TPF release link library. Standard MVS linkage editor procedures can be used to place it there. The table must be named BMGL.

For MVS systems, you must change the program name to IEWL.

The program version number is represented by *nn* (for example, BMP0*nn*).

If all of the DOF modules are not located in the current version of the TPF object library, it may be necessary to concatenate an object library from an earlier release to the OBJLIB data definition.

If the TPF system does not support local 3270 terminals, you do not need to include the BMP5, BMP6, and BMP7 segments in the link edit.

If the TPF system does not support PTV, you do not need to include all the BMP segments in the link edit.

Operating Procedure

The following steps, in the specified sequence, are required to run the offline diagnostic output formatter (DOF).

1. Start the MVS initial program load (IPL) procedure.
2. Submit the link-edited JCL to your JES2 or JES3 subsystem.
3. Mount and ready the RTL/RTA tape on a tape device when you are requested.
4. If you did not specify the DOF options in the JCL EXEC statement, respond the messages that are displayed and specify the appropriate DOF options. See “Specifying DOF Options by Responding to Console Messages” on page 1441 for more information.
5. Mount and ready scratch tapes on the appropriate tape devices when you are requested.
6. Ready the system printer for output.

Output

The output of the offline diagnostic output formatter is a printer listing. This output includes:

- TPF main storage dumps
- PTV input/output message stream
- PTV control and attention messages
- Output from real-time trace (RTT)
- Communication control program trace
- Selective file dump and trace (SFDT)
- Terminal simulation.

After printing the previously listed output, subsequent output depends on the type of PTV run that is used:

- Live input or PTV STV mode (TV)
- PTV package test mode (RS).

TV: Live Input or PTV STV Mode

The diagnostic output formatter (DOF) makes a pass through the PTV input/output message stream by terminal type. The input/output message stream is formatted on the printer according to one of the following terminal types:

- CRTS
- 1977
- 1980-021
- 1980-024
- 3270
- Teletype.

RS: PTV Package Test Mode

The number of passes through the PTV input/output message stream depends on the number of simulation options and the number of terminals to be simulated, as specified at the time that the test unit tape was created. There can be 0–9 passes. The input/output message stream is formatted the same way for TV and RS mode. See *TPF Program Development Support Reference* for more information about the required format of simulation option cards.

Messages

This section provides information about the DOF error messages.

Error Messages

USER ABEND 1111

Explanation: This is an input parameter error that can occur when you specify TV or RS in the JCL EXEC statement. See “Specifying DOF Options in the JCL EXEC Statement” on page 1447 for more information.

USER ABEND 1112

Explanation: This is an input parameter error that can occur when you specify whether or not you want STPP options (Y or N) in the JCL EXEC statement. See “Specifying DOF Options in the JCL EXEC Statement” on page 1447 for more information.

USER ABEND 1113

Explanation: This is an input parameter error that can occur when you specify the processing options in the JCL EXEC statement. See “Specifying DOF Options in the JCL EXEC Statement” on page 1447 for more information.

USER ABEND 2000

Explanation: This error occurs when the trace control table or the trace count output table contains an incorrect logical record length.

The logical record length specified determines how real-time trace (RTT) output is formatted, based on the TPF model or release number, to account for SVC mnemonic differences between TPF releases.

USER ABEND 2222

Explanation: There is a permanent I/O error on the tape. The SYNAD error analysis routine is entered.

USER ABEND 2999

Explanation: A volume sequence error was detected while postprocessing a multivolume dump. An error message is written to the system printer before the ABEND occurred. See *Messages (System Error and Offline)* and *Messages (Online)* for more information about specific errors.

USER ABEND 3333

Explanation: A bad communication control program (CCP) trace record was encountered. The record contains the CT record identifier, but there is no data in the record. The CCP trace cannot be post processed. Run the post processor without the CCP trace (CP) option to process other records that may be on the tape.

USER ABEND 3999

Explanation: An SVC index is not valid. A CFISVC macro could not decode an SVC index. This may be caused by a back-level macro table in CCMCDC.

Note: To receive standard MVS dumps for all ABENDs, code the SYSUDUMP or SYSABEND data definition cards for the JCL; for example:

```
//SYSUDUMP  DD  SYSOUT=A
```

Specifying DOF Options by Responding to Console Messages

You can specify the DOF options by responding to 3 messages that are displayed on the console.

See “Specifying DOF Options in the JCL EXEC Statement” on page 1447 for information about specifying the DOF options in the JCL EXEC statement.

First Message for DOF Options

The following message is displayed to request information about the type of PTV run that was used to generate the RTL tape:

```
IDENTIFY RTL OUTPUT-LIVE OR PTV/STV = TV,
PTV/III = RS
```

Enter one of the following responses:

TV
The RTL tape was generated by PTV in STV mode or from a TPF system without PTV (live input).

RS
The RTL tape was generated by PTV in package test mode.

Note: This message is not displayed if the TPF system was generated without PTV support or if the TPF system was generated with PTV support but without unit or package test support. In these cases, the TV response is assumed.

Second Message for DOF Options

The following message is displayed to determine whether you want to use the default DOF options or specify your own options:

```
STPP OPTIONS WANTED?
YES=Y, NO=N
```

Enter one of the following responses:

N The RTL tape is processed with the following default options:
- Real-time trace (RTT), selective file dump and trace (SFDT), and communication control program (CCP) trace
- PTV input/output message stream
- All simulators active
- Print simulated input/output messages
- Process all PTV package test units (GO option)
- Process all system error dumps (print minidumps)
- Process all snapshot dumps.

Note: You can change these default options by using the system initialization program (SIP). PTV output, including simulation, is not processed if the SIP SYGLB/SYSET indicates no PTV support.

Y A third message is displayed to request the DOF options.

If you entered **TV** in response to the first message (or if the TV response was assumed), see “Third Message for DOF Options (TV Mode)” for more information about the DOF options.

If you entered **RS** in response to the first message, see “Third Message for DOF Options (RS Mode)” on page 1446 for more information about the DOF options.

Third Message for DOF Options (TV Mode)

The following message is displayed to prompt you to enter the DOF options you want to use:

```
SPECIFY OPTIONS
```

You can specify the following DOF options:

CT
Print all communication control program (CCP) trace output.

TR
Print all trace output, which includes output from the real-time trace (RTT) and the selective file dump and trace (SFDT)

IO Print PTV input/output message streams.

SM
Sort PTV input/output messages by the LNIATA/LEID field. This is not available in RS mode. See "First Message for DOF Options" on page 1441 for more information about RS mode.

NOPR
Suppress printing of sorted, simulated PTV input/output messages.

ALL
Does the following:
- Prints all trace output, which includes output from the real-time trace (RTT), the selective file dump and trace (SFDT), and the communication control program (CCP) trace
- Prints PTV input/output message streams
- Sorts input/output messages by the LNIATA/LEID field
- Prints sorted simulated PTV input/output messages
- Prints all system error dumps (minidumps)
- Prints all snapshot dumps
- Activates all simulators.

SI=(ALL)
Activate all simulators.

SI=(*xxxx***)**
Activate the specified simulator, where *xxxx* can be one of the following simulators:
- CRTS
- 1977
- 1980
- (1980-21)
- 1984 (1980-24)
- 3270
- TTYS.

SI=(*xxxx***,***yyyy***,***zzzz***...)**
Activate the specified simulators, which can be any combination of the previously listed simulators.

Note: If the TPF system does not support local 3270 terminals, you cannot specify the SI=(3270) option.

DP=ALLF
Print, in full, all system error dumps.

DP=ALL
Print, in minidump format, all system error dumps.

DP=ALLF-*ssss*
Print, in full, all system error dumps produced from the specified subsystem, where *ssss* is a 1- to 4-character subsystem name.

DP=ALL-*ssss*
Print, in minidump format, all dumps produced from the specified subsystem, where *ssss* is a 1- to 4-character subsystem name.

DP=GOTOF(*xxxx***)**
Go to the first system error dump with dump number *xxxx* and print, in full, all

system error dumps from that point on, regardless of the subsystem that produced them. The variable *xxxx* is a 1- to 6-digit system error dump number. Leading zeros are not necessary.

DP=GOTOF(*xxxx***-***ssss***)**
Go to the first system error dump with dump number *xxxx* from subsystem *ssss* and print, in full, all system error dumps from that point on. The variable *xxxx* is a 1- to 6-digit system error dump number. Leading zeros are not necessary. The variable *ssss* is a 1- to 4-character subsystem name.

DP=GOTO(*xxxx***)**
Go to the first system error dump with dump number *xxxx* and print minidumps of all system error dumps from that point on, regardless of which subsystem produced them. The variable *xxxx* is a 1- to 6-digit system error dump number. Leading zeros are not necessary.

DP=GOTO(*xxxx***-***ssss***)**
Go to system error number *xxxx* that was produced from subsystem *ssss*. Print minidumps of all system error dumps that were produced by this subsystem from that point on. The variable *xxxx* is a 1- to 6-digit system error dump number. Leading zeros are not necessary. The variable *ssss* is a 1- to 4-character subsystem name.

DP=PRINT(*xxxx***F,***yyyy***,***zzzz***...)**
Print only system error dumps numbered *xxxx* (for full dumps), *yyyy*, and *zzzz* (for minidumps), regardless of the subsystem that produced them. The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers.

DP=PRINT(*xxxx***F-***ssss***,***yyyy***-***tttt***,***zzzz***-***uuuu***...)**
Print only system error number *xxxx* from subsystem *ssss* (full dump), system error number *yyyy* from subsystem *tttt*, and so on (minidumps). The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers. The variables *ssss*, *tttt*, and *uuuu* are 1- to 4-character subsystem names.

You can specify as many as 10 dump number and subsystem name pairs.

Note: The last subsystem name that you specify is used as the default subsystem for any subsequent dump numbers that you specified without a subsystem name. For example, if you specified DP=PRINT(XXXX-SSSS,YYYY,ZZZZ-UUUU), dump XXXX of subsystem SSSS, dump YYYY of subsystem SSSS, and dump ZZZZ of subsystem UUUU are printed. If dump number YYYY also exists for another subsystem on the RTL tape, it is not printed.

DP=SKIPF(*xxxx***,***yyyy***,***zzzz***...)**
Print, in full, all system error dumps with the exception of system error dump numbers *xxxx*, *yyyy*, *zzzz...* regardless of the subsystem that produced them. The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers.

You can specify as many as 10 dump numbers.

DP=SKIP(*xxxx***,***yyyy***,***zzzz***...)**
Print minidumps of all system error dumps with the exception of system error dump numbers *xxxx*, *yyyy*, *zzzz...* regardless of the subsystem that produced them. The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers.

You can specify as many as 10 dump numbers.

DP=SKIPF(*xxxx***-***ssss***,***yyyy***-***tttt***,***zzzz***-***uuuu***...)**

Print, in full, all system error numbers with the exception of dump number *xxxx* for subsystem *ssss*, dump number *yyyy* for subsystem *tttt*, dump number *zzzz* for subsystem *uuuu*, and so on.

The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers. The variables *ssss*, *tttt*, and *uuuu* are 1- to 4-character subsystem names.

You can specify as many as 10 dump number and subsystem name pairs.

Note: The last subsystem name that you specify is used as the default subsystem for any subsequent dump numbers that you specified without a subsystem name. For example, if you specified DP=SKIPF(XXXX-SSSS,YYYY,ZZZZ-UUUU), dump XXXX of subsystem SSSS, dump YYYY of subsystem SSSS, and dump ZZZZ of subsystem UUUU are not printed. If dump number YYYY also exists for another subsystem on the RTL tape, the dump for that subsystem ***is*** printed.

DP=SKIP(*xxxx***-***ssss***,***yyyy***-***tttt***,***zzzz***-***uuuu***...)**

Print, in minidump format, all system error numbers with the exception of dump number *xxxx* for subsystem *ssss*, dump number *yyyy* for subsystem *tttt*, dump number *zzzz* for subsystem *uuuu*, and so on.

The variables *xxxx*, *yyyy*, and *zzzz* are 1- to 6-digit system error dump numbers. The variables *ssss*, *tttt*, and *uuuu* are 1- to 4-character subsystem names.

You can specify as many as 10 dump number and subsystem name pairs.

Note: The last subsystem name that you specify is used as the default subsystem for any subsequent dump numbers that you specified without a subsystem name. For example, if you specified DP=SKIP(XXXX-SSSS,YYYY,ZZZZ-UUUU), dump XXXX of subsystem SSSS, dump YYYY of subsystem SSSS, and dump ZZZZ of subsystem UUUU are not printed. If dump number YYYY also exists for another subsystem on the RTL tape, the dump for that subsystem ***is*** printed.

DP=NO

Do not print any system error dumps.

SNAP=ALL

Process all snapshot dumps.

SNAP=ALL-*ssss*

Process all snapshot dumps from subsystem *ssss*, where *ssss* is a 1- to 4-character subsystem.

SNAP=TIME(*hh.mm.ss***-***hh.mm.ss***,....)**

Process all snapshot dumps between the specified starting time and ending time, where *hh.mm.ss* is the timestamp in hours, minutes, and seconds. If you do not specify an ending time, all the snapshot dumps after the starting time are processed.

You can specify as many as 3 pairs of time ranges.

Note: Ensure that the ending time is greater than the starting time. STPP does not handle time specified across a day boundary.

SNAP=PREFIX(*w*,*x*,*y*,*z*)
Process all snapshot dumps with the specified prefix, where *w*, *x*, *y*, and *z* are prefixes.

You can specify as many as 4 prefixes.

SNAP=CODE(*xxxxxxxx*,*yyyyyyyy*,...)
Process all snapshot dumps with the specified snapshot error codes, where *xxxxxxxx* and *yyyyyyyy* are snapshot error codes.

You can specify as many as 10 snapshot error codes.

You can specify combinations of the previous options with the following exceptions:

- The ALL option can be specified only by itself or with the NOPR option.
- The SM option must be specified with an SI option, where the SI option actually specifies the type of messages to be sorted.
- Only one SI option is allowed for each response but can include more than one type of terminal device.
- Only one DP option is allowed for each response. If no DP option is specified, the DP=ALL option is assumed.
- Only one SNAP option is allowed for each response.
- The only option that has a default value is the DP option. Therefore, if you do not specify a particular option, no data is printed for that option.
- Separate each option with a comma (,) and end the last option with a period (.).
- If the TPF system does not support PTV, you cannot specify the IO, SM, NOPR, or SI options. In addition, the associated defaults in the ALL option are not generated when PTV is not supported.
- If you do not specify an option correctly, the entire response is rejected.

Examples of DOF Options (TV Mode)

The following shows examples of the DOF options.

DP=PRINT(19,57,104,175),TR,IO,SI=(ALL).
Print only system error dump numbers 19, 57, 104, and 175. Print all RTT, SFD, and SFT output. Print all of the PTV input/output message stream. Simulate all of the input/output messages.

DP=SKIP(2,4,6,8),CT,SM,SI=(CRTS,1977).
Print all system error dumps except dump numbers 2, 4, 6, and 8. Print all CCP trace output. Sort and print only CRTS and 1977 input/output messages in simulated format.

DP=GOTO(175),SM,NOPR,SI=(1980).
Print system error dump number 175 and all system error dumps following it. Sort, but do not print, the 1980-21 input/output messages. Messages are sorted (LNIATA/LEID sequence) on the device specified by the //SORTOUT DD card of the JCL that is running.

Third Message for DOF Options (RS Mode)

If you entered **RS** in response to the first message, RUNID is displayed with the second message each time the PTV RUNID record is found.

All of the DOF options available in TV mode are also available in RS mode. However, one additional option is available in RS mode:

GO
Allows you to start processing the RTL/RTA tape again from the point where the

option was entered. The remainder of the unprocessed PTV RUNIDs (test units) are processed with the same options that were specified when you specified the GO option.

This feature allows you to process each PTV RUNID (test unit) with different options and output. In addition, the GO option allows you to process the remaining RTL/RTA tape (all test units) with the same options.

Note: The N response to the second message includes the GO option, as does the ALL option for the third message in TV mode.

Specifying DOF Options in the JCL EXEC Statement

You can specify the DOF options in the JCL EXEC statement instead of entering them in response to console messages.

Examples of EXEC Parameters

The following EXEC statements are examples of how you can specify the DOF options by using the PARM= parameter.

```
// EXEC   PGM=PPCP,REGION=250K,
//        PARM=('TV,Y,DP=PRINT(19,57,104,175)','TR,IO,SI=(ALL).')
```

```
// EXEC   PGM=PPCP,REGION=512K,
//        PARM=('TV,Y,DP=SKIP(2,4,6,8),CT,SM,SI=(CRTS,1977).')
```

```
// EXEC   PGM=PPCP,REGION=512K
//        PARM=('TV,Y,DP=GOTO(175),SM,NOPR,SI=(1980).')
```

```
// EXEC  PGM=PPCP,REGION=250K,
//       PARM=('RS,Y,DP=NO,TR.Y'),
//       'DP=PRINT(1045,1046),IO,GO.')
```

The RTL/RTA tape being processed contains multi-test unit output from PTV test units (the RS option). In the first test unit, print only the real-time trace (RTT) and selective file dump and trace (SFDT) output. In the second and all succeeding test units (GO option), print system error dump numbers 1045 and 1046 plus the PTV input/output message stream of the test unit.

```
// EXEC   PGM=PPCP,REGION=250K,PARM='N'
```

The RTL/RTA tape being processed was generated on a TPF system that did not support PTV or that supported PTV only in STV mode. As a result, the TV or RS option is not coded. The N option requests all of the default DOF options. In a system without PTV support, DOF prints all trace output, all CCP trace output, and all system error dumps. In a system where PTV is supported only in STV mode, DOF also prints the PTV input/output message stream plus the output from all generated simulators.

When specifying the DOF options in the EXEC statement, use extreme care with the punctuation that is required by the IBM MVS system and the use of PPCP, especially for multiple card input. Commas (,) required to end a continued card are passed to PPCP as part of the parameter field.

Console Error and Attention Messages

See the IBM MVS message library for an explanation of the console messages that are generated by the IBM MVS system.

Restart Procedures

There is no restart facility. If a program check or an irrecoverable I/O error occurs, you must run the job again.

Hardware Requirements

The following hardware is required:

- 1 tape drive for the RTL/RTA tape
- 1 DASD for the scratch work area (DDSCTH)
- 1 console for job control
- 1 printer for output listings
- 1 DASD for the unit that contains the TPF link library.

Additional DASD for MVS sort/merge may be required if the SM option is requested.

Stand-Alone Dump Utility and Postprocessor

This chapter describes the stand-alone dump (SADUMP) utility and the postprocessor.

Stand-Alone Dump (SADUMP) Utility

The stand-alone dump (SADUMP) utility is an offline utility that can be IPLed from tape. It is used to dump main storage when a normal online dump cannot be processed because of a system *hang* condition.

Setup

The SADUMP utility is assembled into an object library during the system initialization program (SIP). Then, it is loaded to a nonlabeled tape using IEBGENER. See *TPF System Generation* for more information about SIP.

Note: It is recommended that you use the highest format tape device available to the TPF system and to the IBM MVS system to maximize the dump tape capacity. A standard cartridge mounted on a 3480 tape device holds approximately 200 MB of data. An enhanced capacity cartridge system tape (ECCST) mounted on a 3490E tape device holds approximately 800 MB of data. A 3590 cartridge mounted on a 3590 tape device holds 10–40 GB of data.

When the tape is IPLed, the dump is written after the SADUMP program, so the tape can be reused for subsequent dumps without reloading the SADUMP object code.

Sample JCL for Creating an SADUMP Tape

The following is a sample of the JCL that you need to load the SADUMP utility to tape:

```
//SADLOAD EXEC PGM=IEBGENER,REGION=58K
//SYSPRINT  DD SYSOUT=A
//SYSIN     DD DUMMY
//SYSUT1    DD DSN=*OBJECT.LIBRARY*(SADUMP40),DISP=SHR
//SYSUT2    DD DSN=DUMP.TAPE,UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(,NL),
//  VOL=SER=XXXXXX,DCB=(RECFM=FB,LRECL=80,BLKSIZE=80)
```

Note: Replace `*OBJECT.LIBRARY*` with the library containing SADUMP40.

Processing Procedures

If there is a system *hang* condition, follow these procedures:

1. Press the STOP key.
2. Enter **STORE STATUS** from the operator control panel. This should be done from the main I-stream (the CPU from which the TPF system was IPLed).

   Even if you do not enter **STORE STATUS**, the dump is still performed. However, the control registers, current PSW, CPU Timer, and TOD clock comparator values are not captured. The general purpose registers are always captured.
3. Mount and make ready the SADUMP tape on an available tape device. The available tape device must be the same format as the tape device used to create the tape.

4. IPL the tape ***without*** the RESET or CLEAR options. The SADUMP utility then scans for the end of the SADUMP program and begins writing the content of main memory to the SADUMP tape.

Note: The SADUMP tape must be IPLed from the same I-stream that issued the STORE STATUS so that the store status information is formatted properly by the dump postprocessor.

Normal End-of-Job

All communication with the operator is through the current PSW.

Normal End-of-Dump PSW

When the dump ends successfully, the SADUMP tape is rewound and unloaded and the TPF system is placed in a disabled wait state with the following PSW:

X'000A000000000E0D'

If the memory size exceeds the capacity of the SADUMP tape, the first tape is unloaded and the TPF system is placed in an enabled wait state (no PSW displays on the operator control panel) until you mount a second tape on the same tape device. The second tape can be any scratch tape. When the second tape is readied, the dump continues to normal end-of-job status.

Tape Mount Request PSW

If you are in doubt about the status of the dump, stop the processor and display the current PSW. The following PSW indicates that the dump is waiting for a second tape:

X'020A00000000FEED'

Abnormal End-of-Job

If the dump finds an irrecoverable error, the SADUMP tape stops without rewinding or unloading.

I/O Error PSW

The TPF system is placed in a disabled wait state and displays a PSW that contains error information in the following format:

X'000A0000'*xxxxyyyy*

where:

xxxx
: is the operation code of the failing I/O instruction.

yyyy
: is one of the following:

- The condition code of the failing I/O operation in the form of X'0CC'*x*, where *x* is 0, 1, 2, or 3.
- The device/subchannel status from the IRB.

Unexpected System Interrupts

The TPF system is placed in a disabled wait state and displays a PSW that contains error information in the following format:

X'000A0000'*xxxxxxxx*

where:

xxxxxxx
is one of the following PSW codes that describes the interrupt that occurred:

1111111
An external interruption

2222222
A supervisor call interruption

3333333
A program interruption

4444444
A machine check interruption

5555555
An input/output interruption.

Stand-Alone Dump Postprocessor (SADPRT) Utility

The stand-alone dump postprocessor (SADPRT) utility is a utility program that runs under IBM MVS control to create a printed storage dump from the data captured on the SADUMP tape.

The dump output is unformatted; that is, TPF dump tags and formatted storage blocks do not display as they do in an online TPF dump. However, the data is translated into EBCDIC on the right-hand side of each page to assist you with locating items in the dump. The general purpose registers and store status information (if a STORE STATUS was issued before the dump) are displayed at the beginning of the dump.

The address range of main memory that is printed from the dump tape may be limited by optionally specifying the following PARM= parameter in the EXEC statement:

PARM=(*start_address***,***end_address***)**

where:

start_address
is the beginning address of the range to print (8 bytes).

end_address
is the end address of the range to print (8 bytes).

Parameters are positional. You must specify leading zeros if *start_address* or *end_address* is less than 8 characters. If you omit a parameter, you must specify a comma (,) in place of that parameter. If you omit *start_address*, the beginning address range defaults to 0. If you omit *end_address*, the end of the address range defaults to the end of the dump. If you do not specify the PARM parameter, the SADPRT utility prints the entire dump.

The PARM= parameter errors are reported by an ABEND accompanied by one of the following messages:

- PARM= CONTAINS INVALID CHARACTER *x*

  where *x* is a non-hexadecimal character.
- PARM= START ADDRESS NOT LESS THAN END ADDRESS.

- PARM= CONTAINS WRONG LENGTH INPUT. ADDRESSES MUST BE 8 CHARACTERS LONG.
- PARM= ADDRESS GREATER THAN X'7FFFFFFF'.

The following are some examples of how to code the PARM= parameter :

- To print from 40000–AF8000, code:
  `PARM=(00040000,00AF8000)`
- To print from 0–AF8000, code:
  `PARM=(,00AF8000)`
- To print from 40000 to the end of the dump, code:
  `PARM=(00040000)`

Setup

The SADPRT utility is assembled and link-edited during the system initialization program (SIP). The tape device must be able to read the format that was used by the tape device that created the dump. If multiple tape reels were used, they must be coded on the DUMPTAP DD card in the same order that they were created, as shown in "Sample JCL for Processing the SADPRT Utility".

Sample JCL for Processing the SADPRT Utility

The following example shows the JCL required to process the SADPRT utility on an IBM MVS system:

```
//STEP1    EXEC PGM=SADPRT40,REGION=8K,PARM=(001D0E00,003BC000)
//STEPLIB  DD   DSN=*LINK.LIBRARY*,DISP=SHR
//DUMPTAP  DD   DSN=DUMP.TAPE,UNIT=TAPE,DISP=OLD,LABEL=(,NL),
//  VOL=SER=(VOL001,VOL002)
//DUMPPRT  DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
//SYSUDUMP DD   SYSOUT=A
/*
```

Note: Replace `*LINK.LIBRARY*` with the library containing SADPRT40.

Processing Procedures

If you do not specify the PARM= parameter, the entire dump is printed. If the dump was created on multiple tape volumes, the SADPRT utility calls for them in the order that they are specified in the VOL=SER= parameter of the DUMPTAP DD card. Because nonlabeled tapes are used, no internal sequence checking is performed. The tapes ***must*** be mounted in the correct order.

Normal End-of-Job

The dump output is spooled to the SYSOUT class specified on the DUMPPRT DD card. See "Sample JCL for Processing the SADPRT Utility" for more information. The SADPRT utility suppresses duplicate print lines to conserve paper. For example, only the first line of a large storage area containing binary zeros appears in the dump.

Abnormal End-of-Job

The SADPRT utility issues no error messages of its own other than the PARM= error messages that were previously listed. Abnormal conditions are handled and reported by IBM MVS error recovery.

Program Test Vehicle

This chapter describes the procedures that are available for the program test vehicle (PTV), which is an online utility that allows you to simulate parts of the TPF system to test applications in progressive levels of system involvement. You can test applications individually, with related programs, in a transaction, or in a full system.

Note: You can use PTV only in a uniprocessor environment. Enter the ZCNIS command to deactivate all but one I-stream before you start PTV.

PTV Modes

You can run PTV in the following modes:

Mode **Description**

PH3 In phase 3 (package test) mode, more than one test unit can be included on a test unit tape (TUT). A test unit can also span more than one TUT tape. PTV starts automatically when it runs in phase 3 mode. During each test unit and after each test unit is complete, you can enter live input if you specified live input on the RUNID card.

STV In system test vehicle (STV) mode, only one test unit can be included on a TUT tape. However, this test unit can span more than one TUT tape. You start and control PTV in STV mode by entering the ZSTVS commands. You can enter live input during the unit test.

LIVE Live mode stops PTV and returns control to the TPF system.

See *TPF Program Development Support Reference* for a complete description of PTV, the PTV modes, the prerequisites for running PTV, and input to PTV.

Running PTV

Use the following procedures to run PTV.

Setup Procedure

Do the following before you start PTV in phase 3 mode or STV mode.

1. Use the system test compiler (STC) to create a test unit tape (TUT) that contains the test units that you want run. See “System Test Compiler” on page 1477 for more information about the STC.
2. If you are running PTV in PH3 mode, mount and make ready the online disk packs as specified in the *TPF System Installation Support Reference*.
3. Mount and ready one scratch tape for the real-time log tape (RTL) and one scratch tape for the real-time tape (RTA).
4. Mount the get file storage (GFS) directories, if requested.
5. Mount and make ready a pilot system tape (SDF), if requested. This tape must be write protected.
6. Mount and make ready any additional tapes that are required for testing (general tapes).
7. Ready any consoles that will be used for live input during the PTV run.
8. If you are performing a package test (phase 3 mode), mount and make ready a nonblocked database restore (DBR) output tape.

Running PTV in Phase 3 Mode

To run PTV in phase 3 mode, do the following:

1. Enter **ZSTVS TEST RESET** to reset PTV. The TPF system prompts you to perform an initial program load (IPL).
2. Enter the ZRIPL command to perform an IPL.
3. When PTV requests during restart that you specify the test mode for PTV run, enter **ZSTVS TEST YES**.
4. Mount the appropriate tapes as requested by the TPF system.
5. The TPF system automatically cycles to the appropriate state as specified in the RUNID card for the test unit.

   Note: If you are testing in a specific subsystem, the basic subsystem cycles to NORM state and the subsystem in which you are testing cycles to the appropriate state, as specified in the RUNID card for the test unit.
6. If you expect output from the test units to exceed the capacity of one tape reel, mount a standby tape. When you mount a standby tape, tape switching occurs automatically.

   You can mount a standby tape at any time.
7. PTV runs the first test unit on the first TUT tape. If you specified live input on the RUNID card for the test unit, the following message is displayed to indicate that you can enter live input at any time during the test run:

   ```
   PTVP0001I LIVE TEST ACTIVATE LINES
   ```
8. After PTV has entered all the messages in the test unit, the following message is displayed to indicate that the test unit is completed and you can now enter more live input before continuing to the next test unit:

   ```
   TO TERMINATE TEST UNIT DEACTIVATE LINES AND ENTER:
   ZSTVS TEST END LIVE
   ```
9. When you are finished entering live input, enter **ZSTVS TEST END LIVE** to end the live input and start the next test unit.
10. When you reach the end of the TUT tape, the following message is displayed:

    ```
    PTVC0002I TO INITIATE NEXT RUN ENTER ZRIPL
    ```
11. If you want to continue PTV testing from the next TUT tape, do the following:
    a. Dismount the current TUT tape.
    b. Mount and make ready the next TUT tape on the same tape device.
    c. Enter the ZRIPL command to perform an IPL.
    d. Enter the appropriate ZSTVS TEST command when PTV requests during restart that you specify the test mode for PTV run.
12. If you want to end PTV at this time, do the following:
    a. Enter the ZRIPL command to perform an IPL.
    b. Enter **ZSTVS TEST LIVE** when PTV requests during restart that you specify the test mode for the PTV run.

       PTV automatically dismounts the current TUT tape and restores all the programs and data records that were changed by the test run to their original condition before the test run. PTV also dismounts the DBR tape.
    c. Enter **ZTOFF ALL** to dismount the RTL tape, the RTA tape, and any other tapes you were using.

Running PTV in STV Mode

To run PTV in STV mode, do the following:

1. Enter **ZCYCL NORM** to cycle the TPF system to NORM state.
2. Enter **ZSTVS TEST RESET** to reset PTV. The TPF system prompts you to perform an IPL.
3. Enter the ZRIPL command to perform an IPL.
4. When PTV requests during restart that you specify the test mode for the PTV run, enter **ZSTVS TEST STV**.
5. Mount the appropriate tapes, as requested by the TPF system.
6. The TPF system automatically cycles to NORM state.
7. If you expect output from the test runs to exceed the capacity of one tape reel, mount a standby tape. When you mount a standby tape, tape switching occurs automatically.

   You can mount a standby tape at any time.
8. Enter the ZSTVS START command to start the PTV in STV mode. PTV runs the test unit on the TUT tape. You can enter live input at any time during the test run.

Stopping PTV

To stop PTV, do the following:

1. Enter **ZSTVS STOPT** to stop PTV.
2. Enter **ZSTVS TEST RESET** to reset PTV. The TPF system prompts you to perform an IPL and to dismount all active and standby tapes.
3. Enter **ZTOFF ALL** to dismount all the active and standby tapes.
4. Enter the ZRIPL command to perform an IPL.
5. When PTV requests during restart that you specify the test mode for the PTV run, enter **ZSTVS TEST LIVE**.

   PTV automatically dismounts the current TUT tape and restores all the programs and data records that were changed by the test run to their original condition before the test run. PTV also dismounts the DBR tape if it was running in phase 3 mode.
6. The TPF system ends PTV and continues to cycle to the appropriate state.

Changing From One Mode to Another

Do the following to change the PTV mode at any time:

1. Enter **ZSTVS TEST RESET** to reset PTV. The TPF system prompts you to perform an IPL and to dismount all active and standby tapes.

   Note: If you are changing from phase 3 mode to another mode, PTV automatically restores all the programs and data records that were changed during phase 3 testing to their original condition. It also dismounts the DBR tape.
2. Enter **ZTOFF ALL** to dismount all the active and standby tapes.
3. Enter the ZRIPL command to perform an IPL.
4. Enter the appropriate ZSTVS TEST command when PTV asks you during restart to specify the test mode for the PTV run.

Restarting the TPF System After a Catastrophic Error

1. If a catastrophic error occurs during a test, such as a system-detected hardware or software error, the TPF system issues a dump to the RTL tape, an alarm sounds, and the control program error recovery performs an IPL.
2. If PTV is unable to control the tracing of a test unit, the following message is displayed:

   ```
   PTV UNABLE TO TRACE
   ```

   The TPF system takes a main storage dump and goes into wait state. Enter the ZRIPL command to continue testing.
3. If the control program error recovery cannot perform an IPL, you must manually perform an IPL for the prime module.

If it is not possible to recover using this procedure, enter the ZTWTM command to write 2 tape marks on the RTL tape and then dismount it for postprocessing.

Irrecoverable Errors

If the following message is displayed at any time, reload the TPF system as it was before testing began and rerun the test unit:

```
PTV ABORT RELOAD SYSTEM
```

Note: This message is usually displayed when PTV cannot retrieve the PTV keypoint record.

Diagnostic Output Formatter (DOF)

Data from a PTV test run is written to the real-time log tape (RTL) that you mounted and made ready during the setup procedure. This PTV output is formatted and printed by the offline diagnostic output formatter (DOF).

An EOJ record must be the last record on the RTL tape in order for DOF to process the PTV output. This EOJ record is written to the RTL tape when one of the following conditions occurs:

- You enter the ZSTVS STOPT command
- A catastrophic error occurs during an STV run.

See “Diagnostic Output Formatter” on page 1437 for more information about DOF.

System Performance and Measurement

This chapter describes the system performance and measurement package, which consists of the online data collection utility and the offline data reduction utility.

Online Data Collection

The *online data collection utility* gathers detailed and summary performance data on the real-time data collection tape (RTC). You can run 4 different collectors to gather this performance data:

Collector	Description
File collector	Collects file and virtual file address (VFA) performance data.
Message collector	Collects message performance data for resources that are defined to the TPF system when the data collection utility is started. **Note:** Message performance data is not collected for resources that are defined to the TPF system after the data collection utility is started.
Program collector	Collects program performance data.
System collector	Collects a cross section of system performance data.

Use the ZMEAS commands to start, stop, and restart online data collection.

Starting Data Collection

You can run data collection in 2 different modes: continuous mode and sampling mode.

Continuous Mode

In *continuous mode*, all 4 of the collectors are run one after the other to gather summary performance data, and then one collector is run for a specified interval of time to collect detailed performance data. This collection interval is repeated 300 times, which is the default defined by INTVL in DATACO. For example, you can run the system collector in continuous mode and specify that it is run for 5-second intervals. These collection intervals are repeated 300 times for a total of 1500 seconds, or 25 minutes.

Sampling Mode

In *sampling mode*, both summary and detailed performance data are gathered by one or more specified collectors, in turn. These collectors are run one after the other in a specified order for specified collection intervals. You determine if the collection intervals for the different collectors are the same. After the specified collectors are run for the specified collection intervals, 1 *collection cycle* is complete. A *gap*, which you define (and is at least 3 seconds long), follows each collection cycle. Together, the collection cycle and the gap make up a *sampling period*. This sampling period is repeated again and again for a specified length of time.

For example, you can run the system collector and the message collector in sampling mode. If you specify that the collection interval for the system collector is 20 seconds, the collection interval for the message collector is 15 seconds, and the sampling period is 40 seconds, the length of time for the collection cycle is 35 seconds (20 seconds + 15 seconds) and the length of time for the gap is 5 seconds (40 seconds − 35 seconds). If you specify an overall length of time of 8 minutes (480 seconds), the sampling period is repeated 12 times (480 seconds ÷ 40 seconds).

Specifying How Many Events to Sample

The file and program collectors gather detailed performance data by sampling file and program events. These events can occur very frequently; so frequently, in fact, that the resulting volume of performance data can be a problem. To control the volume of data, you can specify how many events the file and program collectors sample. For example, these collectors can sample every event or every other event.

When you specify this value using the ZMEAS command, you actually specify the number of events to *skip*. That is, if you want the file and program collectors to sample every event, you specify that you want to skip 0 events. If you want these collectors to sample every other event, you specify that you want to skip one event. By default, the file and program collectors skip every 99 events.

It may be easier to think about the number of events that are skipped in terms of a percentage. For example, data collection can skip 50% of the events or it can skip 99% of the events. The following table provides some common percentages for the corresponding skip factor.

Percentage	Skip Factor	Description
0%	0000	Do not skip any events.
50%	0001	Skip every other event.
75%	0003	Sample an event, and then skip 3 events.
80%	0004	Sample an event, and then skip 4 events.
90%	0009	Sample an event, and then skip 9 events.
99%	0099	Sample an event, and then skip 99 events.
99.9%	0999	Sample an event, and then skip 999 events.
99.99%	9999	Sample an event, and then skip 9999 events.

If you know the percentage of events that you want to skip and need to determine the skip factor, use the following formula:

```
skip = (percentage ÷ (100 − percentage))
```

For example, the skip factor for 80% is determined as follows:

```
skip = (80 ÷ (100 − 80))
skip = (80 ÷ 20)
skip = 4
```

If you know the number of events you need to skip and want to determine the percentage, use the following formula:

```
percentage = (skip ÷ (skip + 1)) × 100
```

For example, the percentage for a skip factor of 3 is determined as follows:

```
percentage = (3 ÷(3 + 1)) × 100
percentage = (3 ÷ 4) × 100
percentage = (.75) × 100
percentage = 75
```

To Start Data Collection

1. Enter the ZDSYS command to verify that the TPF system is above 1052 state and not cycling.
2. If the TPF system is not above 1052 state, enter the ZCYCL command to change to the appropriate system state.
3. Enter the ZTVAR and ZTMNT command to mount an active RTC tape.

   You may also want to mount a standby RTC tape if data collection will be running for a long period of time or if there is a significant amount of system activity. You can mount the active or standby RTC tape as blocked or unblocked.
4. Enter the ZMEAS command to start data collection. See “ZMEAS–Start Data Collection” on page 774 for more information about the ZMEAS command.

Stopping Data Collection

Data collection stops automatically and dismounts the RTC tape after the duration you specified in the ZMEAS command is reached. In continuous mode, the duration is equal to the time interval you specified multiplied by the value of INTVL, which is 300 by default. In sampling mode, the duration is equal to the number of minutes you specified.

You can also enter the ZMEAS END command to stop data collection before this duration is complete. If you enter the ZMEAS END command while data collection is running in continuous mode, data collection is not stopped until after the current time interval is completed. If you enter the ZMEAS END command while data collection is running in sampling mode, data collection is not stopped until after the current sampling period is completed.

See “ZMEAS END–End Data Collection” on page 777 for more information about the ZMEAS END command.

Restarting Data Collection

If a *catastrophic* system error occurs while data collection is running and you perform an initial program load (IPL) for the TPF system, do the following to restart data collection:

1. Enter the ZTOFF command to dismount the RTC tape.
2. Enter the ZTMNT command to mount the RTC tape again.
3. Enter the ZMEAS command to start data collection again.

Note: Do not use the performance data gathered before the system error occurred.

If a *non-catastrophic* system error occurs while data collection is running, you may need to enter the ZMEAS RESET command to clear the residual status before you can start data collection again. In this case, do the following:

1. Enter the ZMEAS RESET command to reset data collection. This function automatically dismounts the RTC tape.
2. Enter the ZTMNT command to mount the RTC tape again.

3. Enter the ZMEAS command to start data collection again.

Enter the ZMEAS RESET command only in place of a ZRIPL command. ***Do not*** enter the ZMEAS RESET command to reset data collection if data collection stalls because the system task dispatcher is shut down.

Offline Data Reduction

The *offline data reduction utility* reads the performance data that was gathered on the RTC tape (which has a data definition name of RTC) during data collection and summarizes this performance data to produce various reports. The format and content of the different reports are defined by specifying the various data reduction options in the MVS JCL that is used to run data reduction. These data options have a data definition name of OPTIONS.

When data reduction is completed, each RTC tape is automatically dismounted.

Data Reduction Libraries

The STEPLIB card in the data reduction JCL refers to the library that contains the offline data reduction load module. The data set name of this library is ACP.LINK.REL*nn*.

The SORTLIB card in the data reduction JCL refers to the library that contains the sort and merge program used for data reduction. The data set name of this library is SYS1.LINKLIB.

Starting Data Reduction

Data reduction is started on the MVS system using the appropriate offline JCL. See "Sample JCL for Offline Data Reduction" on page 1467 for an example of this JCL. When you start data reduction, it is assumed that the MVS system is up and running and that the pack where the executable load module resides (ACP.LINK.REL*nn*) is online.

The region size required for running data reduction varies depending on different factors, including the data reduction reports that are produced. You should note the amount of storage used by the MVS system for the various types of data reduction reports and adjust the region parameter accordingly for subsequent reductions.

To Start Data Reduction

1. Define the necessary MVS JCL for data reduction. See "Sample JCL for Offline Data Reduction" on page 1467 for an example of this JCL.
2. Read in the control and option cards.
3. Mount the RTC tapes.

Data Reduction Reports

The format and content of the different data reduction reports are defined by specifying the various data reduction options in the MVS JCL that is used to run data reduction. See "Data Reduction Options" on page 1461 for more information about these options.

Data reduction reports are printed in the order that you defined them in the MVS JCL (DREPT through SYSOUT). A list of the limitations placed on the data reduction programs by the preprocessor variables is displayed at the end of the data reduction reports. This list is ended by a row of asterisks (*). Any error messages that are created while reading the RTC tape and reducing the data are displayed after the row of asterisks. Review this section of the data reduction report to determine if data reduction completed successfully.

Data Reduction Options

Table 14 describes the options you can specify in the data reduction JCL to define the content and format of the reduction reports that are produced.

See “Sample JCL for Offline Data Reduction” on page 1467 for an example of the data reduction JCL.

Table 14. Data Reduction Options

Reduction Keyword	Option	Suboption	Description of Report Produced
CONTROL	DUMP		When no suboptions are specified, this option formats all of the non-data collection records on the RTC tape and includes them in the status report.
CONTROL	DUMP	*nnnn*	This option/suboption combination formats the non-data collection records on the RTC tape (up to the number specified) and includes them in the status report.
SYSTEM			System summary reports only.
SYSTEM	DISTRIBUTION		Statistical distribution of system parameters.
SYSTEM	PLOT		Chronological plot of system parameters.
FILE			File summary reports only. Must specify subsystem for MDBF environment.
FILE	ACCESSESPERID	*nnnn*	All accesses by ID for device *nnnn* (for example, 01A5).
FILE	ACCESSESPERID	S*yyyy*	All accesses by ID for symbolic device *yyyy* (for example, 0004).
FILE	ACCESSESPERID	*nn*	Counts of accesses by ID for each device on channel *nn* (for example, 01).
FILE	ACCESSESPERID	SUM	A summary of accesses by ID for each device type in the system.
FILE	ACCESSESPERID	ALL	A summary and all devices accessed.
FILE	CACHE		When no suboptions are specified, this option selects by default the set of 3990 storage control device summary reports (1 report per active cache subsystem). Each report contains: • One column of totals for all of the active cache subsystems • One column of subtotals for the specific active cache subsystem • One column of statistics for each of the active DASDs backing the cache subsystem.

Table 14. Data Reduction Options (continued)

Reduction Keyword	Option	Suboption	Description of Report Produced
FILE	CACHE	DEVICESUM	This option/suboption combination selects the set of 3990 storage control device summary reports (1 report per active cache subsystem). Each report contains: • One column of totals for all of the active cache subsystems • One column of subtotals for the specific active cache subsystem • One column of statistics for each of the active DASDs backing the cache subsystem.
FILE	CACHE	DEVICEALL	This option/suboption combination selects the following reports: • The 3990 storage control cache summary report, which consists of one column of totals for all of the active cache subsystems and one column of subtotals for each of the active cache subsystems. • The 3990 storage control report, which contains pages of totals for all of the active cache subsystems. • The set of 3990 storage control reports, which consists of subtotals reports, one report per active cache subsystem, each of which contains 2 pages of subtotals for the specific cache subsystem. • The set of 3990 storage control device summary reports, one report per active cache subsystem. Each report contains one column of totals for all of the active cache subsystems, one column of subtotals for the specific active cache subsystem, and one column of statistics for each of the active DASDs backing the cache subsystem. • The set of 3990 storage control reports, one per active DASD backing an active cache subsystem, each of which contains 2 pages of statistics for the specific DASD.
FILE	CACHE	D*nnnn*	This option/suboption combination selects the 3990 storage control report for the specified DASD, which contains 2 pages of statistics for the specific DASD. *nnnn* is the symbolic device address of the DASD.
FILE	CACHE	CACHESUM	This option/suboption combination selects the 3990 storage control cache summary report, which contains one column of totals for all of the active cache subsystems and one column of subtotals for each of the active cache subsystems.

Table 14. Data Reduction Options (continued)

Reduction Keyword	Option	Suboption	Description of Report Produced
FILE	CACHE	CACHEALL	This option/suboption combination selects the following reports: • The 3990 storage control cache summary report, which contains one column of totals for all of the active cache subsystems and one column of subtotals for each of the active cache subsystems • The 3990 storage control report, which contains 2 pages of totals for all of the active cache subsystems • The set of 3990 storage control reports, which contains subtotals reports, one report per active cache subsystem, each of which contains 2 pages of subtotals for the specific cache subsystem.
FILE	CACHE	C*nnnn*	This option/suboption combination selects the 3990 storage control reports, which contains subtotals reports for the specified cache subsystem. Each report contains 2 pages of subtotals for the specific cache subsystem. *nnnn* is the identifier of the cache subsystem.
FILE	CACHE	SALL	This option/suboption combination selects the following reports: • The 3880 storage control report, which contains totals for all of the active 3880 storage directors • The set of 3880 storage control report, which contains subtotals reports, one for each active 3880 storage director, each of which contains subtotals for a specific 3880 storage director. • The set of 3880 storage control reports, 1 for each active DASD backing an active 3880 storage director, which contains statistics for a specific DASD.
FILE	CACHE	S*nn*	This option/suboption combination selects the 3880 storage control report, which is a subtotals report for the specified 3880 storage director that contains subtotals for the specific 3880 storage director. *nn* is the identifier of the storage director.
FILE	COMPARISON		The total accesses for all devices are plotted on histograms by device type.
FILE	CYLINDERANALYSIS	*nnnn*	Counts of accesses by cylinder for device *nnnn* (for example, 01A5).
FILE	CYLINDERANALYSIS	S*yyyy*	Counts of accesses by cylinder for symbolic device *yyyy* (for example, 0004).
FILE	CYLINDERANALYSIS	*nn*	Counts of accesses by cylinder for each device on channel *nn* (for example, 01).
FILE	CYLINDERANALYSIS	SUM	A summary of accesses by cylinder for each nonfixed head file type in the system.

Table 14. Data Reduction Options (continued)

Reduction Keyword	Option	Suboption	Description of Report Produced
FILE	CYLINDERANALYSIS	ALL	A summary and all devices accessed.
FILE	DISTRIBUTION	*nnnn*	Statistical distribution of device *nnnn* (for example, 01A5).
FILE	DISTRIBUTION	S*yyyy*	Statistical distribution of symbolic device *yyyy* (for example, 0004).
FILE	DISTRIBUTION	*nn*	Statistical distribution of all the devices on channel *nn* (for example, 01). See notes.
FILE	DISTRIBUTION	ALL	Statistical distribution of all devices in the system. See notes.
FILE	PATHACTIVITY		Summary of channel path activity exceptions. This report shows discrepancies in channel path use if any exist.
FILE	PLOT	*nnnn*	Chronological plot of device *nnnn* (for example, 01A5).
FILE	PLOT	S*yyyy*	Chronological plot of symbolic device *yyyy* (for example, 0004).
FILE	PLOT	*nn*	Chronological plot of all devices on channel *nn* (for example, 01). See notes.
FILE	PLOT	ALL	Chronological plot of all devices in the system. See notes.
PROGRAM			When no other program-reduction report is requested, this option selects by default the Program Names and Program Enters report for the program selected for reduction.
PROGRAM	CUMULATIVECUTOFFS	*nnn mmm*	The *nnn* value specifies the cumulative percentage at which to cut off the program enters report. The *mmm* value specifies the cumulative percentage at which to cut off the program on-file enters report.
PROGRAM	LEGEND		This option selects the program legend report.
PROGRAM	PACKAGE	*yyyyyyyy*	The *yyyyyyyy* value specifies the name of a package of programs in the program selected for reduction. It may be followed by as many as 5 program name patterns. For example, a pattern of A means any program name beginning with A; *A* means any program name with A as its second or third character; *A means any program name ending in A; and AAAA means that program name.
PROGRAM	PACKAGEREPORTS		When no suboptions are specified, this option selects by default all 3 of the package reports (the package details report, the cumulative package enters report, and the cumulative package on-file enters report) for the program selected for reduction.
PROGRAM	PACKAGEREPORTS	DETAILS	This option/suboption combination selects the package details report for the program selected for reduction.

Table 14. Data Reduction Options (continued)

Reduction Keyword	Option	Suboption	Description of Report Produced
PROGRAM	PACKAGEREPORTS	ENTERS	This option/suboption combination selects the cumulative package enters report for the program selected for reduction.
PROGRAM	PACKAGEREPORTS	ONFILES	This option/suboption combination selects the cumulative package on-file enters report for the program selected for reduction.
PROGRAM	PROGRAMREPORTS		When no suboptions are specified, this option selects by default all 4 of the program reports (the program details report, the cumulative program enters report, the cumulative program on-file enters report, and the program-category statistics report) for the program selected for reduction, or for each package of programs in that program.
PROGRAM	PROGRAMREPORTS	DETAILS	This option/suboption combination selects the program details report for the program selected for reduction, or for each package of programs.
PROGRAM	PROGRAMREPORTS	ENTERS	This option/suboption combination selects the cumulative program enters report for the program selected for reduction, or for each package of programs.
PROGRAM	PROGRAMREPORTS	ONFILES	This option/suboption combination selects the cumulative program on-file enters report for the subsystem selected for reduction, or for each package of programs.
MESSAGE			Action code and message summary reports only.
MESSAGE	ACTIONCODE	*cty appl*	The messages for application *appl* will be collected by action code for *cty* (for example, LAX-LOS ANGELES).
MESSAGE	ACTIONCODE	LINE*ln appl*	The messages for application *appl* will be collected by action code for line *ln* (for example, 05).
MESSAGE	ACTIONCODE	LNIA*lnia appl*	The messages for application *appl* will be collected by action code for line *ln*, terminal interchange *ia* (for example, 0502).
MESSAGE	ACTIONCODE	ALL *appl*	Output message data will be included for application *appl* in the standard action code summary (continuous mode only).
MESSAGE	APPLICATION	*appl*	The messages will be collected by application for *appl* (for example, RES0). This option is applicable to continuous mode only.
MESSAGE	CITYSUMMARY		The activity for each city in the line network.
MESSAGE	DISTRIBUTION	LINE*ln*	Statistical distribution of parameters for line *ln* (for example, 05).
MESSAGE	DISTRIBUTION	ALL	Statistical distribution of parameters for the communication system.
MESSAGE	PLOT	LINE*ln*	Chronological plot of parameters for line *ln* (for example, 05).

Table 14. Data Reduction Options (continued)

Reduction Keyword	Option	Suboption	Description of Report Produced
MESSAGE	PLOT	ALL	Chronological plot of parameters for the communication system.
MESSAGE	STREAM	*cty*	The message stream from *cty* (for example, LAX-LOS ANGELES) will be printed. The output messages will be limited to 65 bytes.
MESSAGE	STREAM	*cty*A	The message stream from *cty* (for example, LAX-LOS ANGELES) will be printed. The output messages will be printed in their entirety as seen by the agent.
MESSAGE	TERMINALACTIVITY	*cty*	The activity for each terminal in *cty* (for example, CHI-CHICAGO) will be reported.
SNA			Action Code, Message Summary, and NCP/ALS/CTC summary reports only.
SNA	DISTRIBUTION	*node*	Statistical distribution of NCP/ALS/CTC parameters for *node* (for example, CTC4076).
SNA	DISTRIBUTION	ALL	Statistical distribution of NCP/ALS/CTC parameters for the system.
SNA	LUACTIVITY	*node**	The activity for each LU requested (for example, LN2C*) will be reported.
SNA	PLOT	*node*	Chronological plot of NCP/ALS/CTC parameters for *node* (for example, NCP2242).
SNA	PLOT	ALL	Chronological plot of NCP/ALS/CTC for the system.
SNA	STREAM	*node*	The message stream from a logical unit (LU) will be printed. The output messages will be limited to 65 bytes.
SNA	STREAM	DDM	The distributed data management (DDM) message stream between the TPF system and the attached database will be printed. The message existence time will be listed for each input message immediately preceded by an output message.
SNA	STREAM	*node* FULL	The message stream from a logical unit (LU) will be printed. The output messages will be printed in their entirety.
REDUCE	SS	*xxxx*	Reduction of all subsystem users for this subsystem, where *xxxx* is subsystem mnemonic.
REDUCE	SYS		Reduction of non-MDBF system.
ALIAS	S *ss*	*name*	The ALIAS cards are free format, where: *ss* is the subsystem mnemonic *ssu* is the subsystem user mnemonic *name* is a 25-character name that starts from the first nonblank character after the subsystem or subsystem user and continues for the next 25 characters, including blanks.
ALIAS	U *ssu*	*name*	

Notes:

1. The collector must be specified before using any options or suboptions. For example, to obtain a summary of accesses by ID for each device type in the system, you must provide a card to specify FILE before the card that specifies FILE ACCESSESPERID SUM.
2. The option cards can be free format because only one space is necessary between the various fields. As many as 8 suboptions can be included on one card.
3. The following options and suboptions can be abbreviated:

Option or Suboption	**Abbreviation**
APPLICATION	APPL
COMPARISON	COMPAR
CYLINDERANALYSIS	CYL
DISTRIBUTION	DIST
PATHACTIVITY	PATH
ACCESSPERID	RID
SYSTEM	SYS
TERMINALACTIVITY	TERMACTIVITY

4. The collector name can never be abbreviated.
5. The ACTIONCODE options are applicable only when RES0 or a comparable user-defined action code application is in the system.
6. If a channel option is chosen on FILE PLOT or FILE DIST, individual reports on other devices (DIST or PLOT) are produced.
7. SNA LUACTIVITY *node** allows you to specify global file-name characters for node names. You can specify 0–7 characters followed by an asterisk (*). All nodes that start with the specified string are selected for the LUACTIVITY report.
8. The reduce card is required at all times.

Sample JCL for Offline Data Reduction

The following JCL is an example of the JCL that you can use to run data collection. Notice the STEPLIB card that defines the library containing data reduction load module. Also notice the SORTLIB card that defines library containing the sort program.

After the RTC tape is defined in the JCL, the various data sets required for sorting and for the data reduction reports are defined. The option cards that specify the types of reports are placed after these data definitions. See “Data Reduction Options” on page 1461 for more information about these option cards.

As with any example, you may need to adapt portions of this JCL to your site requirements.

```
//REDUCE   JOB (82FIV,     ),MSGLEVEL=(1,1),CLASS=A,TIME=1440
//SYSPRGO  EXEC PGM=DATAREAD,REGION=3000K
//STEPLIB  DD   DSN=VIR0000.DEVP.TEST.LK,DISP=SHR
//SORTLIB  DD   DSN=SYS1.SORTLIB,DISP=SHR
//*
//* THE RTC TAPE LABEL STATEMENT MAY NEED TO BE CHANGED DEPENDING
//* UPON THE METHOD THE TAPE WAS INITIALIZED DURING COLLECTION.
//RTC      DD   UNIT=3480,DISP=(SHR,KEEP),DSN=RTC.TAPE,LABEL=(,SL),
//              VOL=SER=(AL0314,AL0768,AL0256),DCB=OPTCD=B
//*
//* THE FOLLOWING DATA SETS ARE USED TO OUTPUT THE
//* DATA REDUCTION REPORTS.
//DREPT    DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//SREPT    DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//FSSUM    DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//PREPT    DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//FILREPT  DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//MOUT     DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
//STATUS   DD   DCB=(RECFM=VBM,LRECL=137,BLKSIZE=141),SYSOUT=A,
//              OUTLIM=50000
```

Figure 2. Data Reduction Sample JCL (Part 1 of 5)

```
//*
//* THE FOLLOWING DATA SETS ARE USED TO LIMIT MVS SYSTEM OUTPUT
//*
//SYSABEND DD   SYSOUT=A,OUTLIM=100000
//SYSPRINT DD   SYSOUT=A,OUTLIM=100000
//SYSOUT   DD   SYSOUT=A,OUTLIM=100000
//PLIDUMP  DD   SYSOUT=A,OUTLIM=100000
//SYSUDUMP DD   SYSOUT=A,OUTLIM=100000
//*
//* THE FOLLOWING DATA SET IS USED BY THE DRIVER FOR THE HEADER
//*
//HEADER   DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),
//              DCB=(RECFM=VM,LRECL=137,BLKSIZE=141),DSN=&&HEADER1
//*
//* THE FOLLOWING DATA SETS ARE USED BY SYSTEM REDUCTION FOR DETAIL
//*       REPORTS
//*
//STEMP    DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&STEMP1,
//              DCB=(RECFM=FB,LRECL=424,BLKSIZE=4240)
//STEMPIS  DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&STEMPIS1,
//              DCB=(RECFM=FB,LRECL=2048,BLKSIZE=10240)
//STEMPID  DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&STEMPID1,
//              DCB=(RECFM=FB,LRECL=1408,BLKSIZE=14080)
//MDWKIN   DD   UNIT=SYSDA,SPACE=(CYL,(2,2)),DISP=(NEW,PASS),
//              DCB=(RECFM=F,LRECL=26,BLKSIZE=26),DSN=&&MDI
//MDWKOUT  DD   UNIT=SYSDA,SPACE=(CYL,(2,2)),DISP=(NEW,PASS),
//              DCB=(RECFM=F,LRECL=26,BLKSIZE=26),DSN=&&MDO
//MDWKWK01 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&MDWK1,
//              DISP=(NEW,DELETE)
//*
//* THE FOLLOWING DATA SETS ARE USED BY FILE REDUCTION FOR DETAIL
//*       REPORTS AND SORTING
//*
//FILSAV4  DD   UNIT=SYSDA,SPACE=(CYL,(9,9)),DSN=&&FILS,
//              DCB=(RECFM=U,LRECL=25000,BLKSIZE=25000)
//FSRTIN   DD   UNIT=SYSDA,SPACE=(CYL,(9,9)),DSN=&&FSRTI,
//              DCB=(RECFM=FB,LRECL=44,BLKSIZE=4312)
//FSRTOUT  DD   UNIT=SYSDA,SPACE=(CYL,(9,9)),DSN=&&FSRTO,
//              DCB=(RECFM=FB,LRECL=44,BLKSIZE=4312)
//FSRTWK01 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK1,
//              DISP=(NEW,DELETE)
//FSRTWK02 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK2,
//              DISP=(NEW,DELETE)
//FSRTWK03 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK3,
//              DISP=(NEW,DELETE)
//FSRTWK04 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK4,
//              DISP=(NEW,DELETE)
//FSRTWK05 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK5,
//              DISP=(NEW,DELETE)
//FSRTWK06 DD   UNIT=SYSDA,SPACE=(TRK,(200),,CONTIG),DSN=&&WORK6,
//              DISP=(NEW,DELETE)
```

Figure 2. Data Reduction Sample JCL (Part 2 of 5)

```
//*
//* THE FOLLOWING DATA SETS ARE USED BY MESSAGE REDUCTION FOR DETAIL
//*       REPORTS AND SORTING
//*
//MPFDF    DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&MPF
//*             DCB=(RECFM=FB,LRECL=x,BLKSIZE=y)
//*
//*  NOTE THAT MPFDF IS A FIXED-BLOCK DATA SET WHOSE LRECL AND
//*  BLKSIZE ARE INSTALLATION DEPENDENT.  THE LRECL CAN BE
//*  DETERMINED FROM THE VALUE OF THE MLSIZE PARAMETER IN
//*  JPC0.  THE BLKSIZE IS UP TO THE INSTALLATION AND SHOULD
//*  BE AN INTEGRAL MULTIPLE OF THE LRECL.
//*
//MTRTIN   DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&MTRTI,
//              DCB=(RECFM=FB,LRECL=20,BLKSIZE=2000)
//MTRTOUT  DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&MTRTO,
//              DCB=(RECFM=FB,LRECL=20,BLKSIZE=2000)
//MTRTWK01 DD   VOL=REF=*.FSRTWK01,DSN=&&WORK1,DISP=(OLD,PASS)
//MTRTWK02 DD   VOL=REF=*.FSRTWK02,DSN=&&WORK2,DISP=(OLD,PASS)
//MTRTWK03 DD   VOL=REF=*.FSRTWK03,DSN=&&WORK3,DISP=(OLD,PASS)
//MTRTWK04 DD   VOL=REF=*.FSRTWK04,DSN=&&WORK4,DISP=(OLD,PASS)
//MTRTWK05 DD   VOL=REF=*.FSRTWK05,DSN=&&WORK5,DISP=(OLD,PASS)
//MTRTWK06 DD   VOL=REF=*.FSRTWK06,DSN=&&WORK6,DISP=(OLD,PASS)
//MARTIN   DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MARTI,
//              DCB=(RECFM=FB,LRECL=26,BLKSIZE=2600)
//MARTOUT  DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MARTO,
//              DCB=(RECFM=FB,LRECL=26,BLKSIZE=2600)
//MSRTIN   DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MSRTI,
//              DCB=(RECFM=FB,LRECL=381,BLKSIZE=2286)
//MSRTOUT  DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MSRTO,
//              DCB=(RECFM=FB,LRECL=381,BLKSIZE=2286)
//MSRTWK01 DD   VOL=REF=*.FSRTWK01,DSN=&&WORK1,DISP=(OLD,PASS)
//MSRTWK02 DD   VOL=REF=*.FSRTWK02,DSN=&&WORK2,DISP=(OLD,PASS)
//MSRTWK03 DD   VOL=REF=*.FSRTWK03,DSN=&&WORK3,DISP=(OLD,PASS)
//MSRTWK04 DD   VOL=REF=*.FSRTWK04,DSN=&&WORK4,DISP=(OLD,PASS)
//MSRTWK05 DD   VOL=REF=*.FSRTWK05,DSN=&&WORK5,DISP=(OLD,PASS)
//MSRTWK06 DD   VOL=REF=*.FSRTWK06,DSN=&&WORK6,DISP=(OLD,PASS)
//MORTIN   DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&MTRTI,
//              DCB=(RECFM=FB,LRECL=51,BLKSIZE=3672)
//MORTOUT  DD   UNIT=SYSDA,SPACE=(CYL,(3,3)),DSN=&&MTRTO,
//              DCB=(RECFM=FB,LRECL=51,BLKSIZE=3672)
//MORTWK01 DD   VOL=REF=*.FSRTWK01,DSN=&&WORK1,DISP=(OLD,PASS)
//MORTWK02 DD   VOL=REF=*.FSRTWK02,DSN=&&WORK2,DISP=(OLD,PASS)
//MORTWK03 DD   VOL=REF=*.FSRTWK03,DSN=&&WORK3,DISP=(OLD,PASS)
//MORTWK04 DD   VOL=REF=*.FSRTWK04,DSN=&&WORK4,DISP=(OLD,PASS)
//MORTWK05 DD   VOL=REF=*.FSRTWK05,DSN=&&WORK5,DISP=(OLD,PASS)
//MORTWK06 DD   VOL=REF=*.FSRTWK06,DSN=&&WORK6,DISP=(OLD,PASS)
```

Figure 2. Data Reduction Sample JCL (Part 3 of 5)

```
//DDMMIN   DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MSRTI,
//              DCB=(RECFM=FB,LRECL=381,BLKSIZE=2286)
//DDMMOUT  DD   UNIT=SYSDA,SPACE=(CYL,(4,3)),DSN=&&MSRTO,
//              DCB=(RECFM=FB,LRECL=381,BLKSIZE=2286)
//DDMMWK01 DD   VOL=REF=*.FSRTWK01,DSN=&&WORK1,DISP=(OLD,PASS)
//DDMMWK02 DD   VOL=REF=*.FSRTWK02,DSN=&&WORK2,DISP=(OLD,PASS)
//DDMMWK03 DD   VOL=REF=*.FSRTWK03,DSN=&&WORK3,DISP=(OLD,PASS)
//DDMMWK04 DD   VOL=REF=*.FSRTWK04,DSN=&&WORK4,DISP=(OLD,PASS)
//DDMMWK05 DD   VOL=REF=*.FSRTWK05,DSN=&&WORK5,DISP=(OLD,PASS)
//DDMMWK06 DD   VOL=REF=*.FSRTWK06,DSN=&&WORK6,DISP=(OLD,PASS)
//*
//* THE FOLLOWING DATA SET IS USED FOR SNA MESSAGE REDUCTION
//*
//SNPFDF   DD   UNIT=SYSDA,SPACE=(CYL,(3,3))
//*
//*
//* THE FOLLOWING DATA SET IS THE OPTION CARDS TO CONTROL THE DATA
//*        REDUCTION PACKAGE
//OPTIONS  DD   *
           PLACE OPTION CARDS HERE
*
SYSTEM
SYSTEM PLOT
SYSTEM DIST
*
FILE
FILE COMPARISON
FILE PATHACTIVITY
FILE PLOT DEVICE
FILE PLOT SDEVICE
FILE PLOT ALL
FILE DIST DEVICE
FILE DIST SDEVICE
FILE DIST ALL
FILE RID DEVICE
FILE RID SDEVICE
FILE RID SUM
FILE RID ALL
FILE CYL DEVICE
FILE CYL SDEVICE
FILE CYL SUM
FILE CYL ALL
FILE CACHE
FILE CACHE SSD
FILE CACHE SALL
FILE CACHE DDEVICE
FILE CACHE DEVICESUM
FILE CACHE DEVICEALL
FILE CACHE CCACHE
FILE CACHE CACHESUM
FILE CACHE CACHEALL
```

Figure 2. Data Reduction Sample JCL (Part 4 of 5)

```
*
PROGRAM
PROGRAM LEGEND
PROGRAM PACKAGE P1 JC*
PROGRAM PACKAGE P2 *
PROGRAM CUMULATIVECUTOFFS 0 0
PROGRAM PROGRAMREPORTS
PROGRAM PROGRAMREPORTS DETAILS
PROGRAM PROGRAMREPORTS ENTERS
PROGRAM PROGRAMREPORTS ONFILES
PROGRAM PACKAGEREPORTS
PROGRAM PACKAGEREPORTS DETAILS
PROGRAM PACKAGEREPORTS ENTERS
PROGRAM PACKAGEREPORTS ONFILES
*
MESSAGE
MESSAGE PLOT ALL
MESSAGE DIST ALL
MESSAGE ACTIONCODE ALL
MESSAGE CITYSUMMARY
MESSAGE STREAM CITY
MESSAGE TERMINALACTIVITY CITY
MESSAGE APPLICATION APPL
*
SNA
SNA PLOT ALL
SNA DIST ALL
SNA LUACTIVITY *
SNA STREAM DDM
SNA LUACTIVITY NODE
SNA STREAM NODE
*
REDUCE    SS BSS
ALIAS S BSS   BSS    SUBSYSTEM
ALIAS S WP    WP     SUBSYSTEM
ALIAS S BART  BART   SUBSYSTEM
ALIAS U HPN   BSS    SSU  ONE
ALIAS U WP1   WP     SSU  ONE
ALIAS U WP2   WP     SSU  TWO
ALIAS U WP3   WP     SSU  THREE
ALIAS U OMEG  BART   SSU  ONE
ALIAS U TF    BART   SSU  TWO
/*
//
```

Figure 2. Data Reduction Sample JCL (Part 5 of 5)

Data Reduction Temporary Files

A number of temporary files are created during data reduction.

Sort Files Created for Input and Output

The following files are created by the sort program and used for input and output.

Note: The definition of each sort output file must be the same as its corresponding input file. See “Sample JCL for Offline Data Reduction” on page 1467 for an example.

File	Description
FSRTIN	Created in JRA2 and passed as input to the sort program by JRA3. This file contains one 40-byte record for every file access recorded for the optioned devices.
FSRTOUT	Created by the sort program and used as input in JRF5.
MARTIN	Created in JRM2 and passed as input to the sort program by JRM2. This file contains one 26-byte record for every message recorded on the RTC tape, or one 26-byte record for every message for the optioned lines, LNIAS, or CITYS.
MARTOUT	Created by the sort program and used as input in JRM4 for continuous mode action code reports.
MDWKIN	Created in JRA2 and passed as input to the sort program by JRS3. This file contains one 26-byte record for every MPIF path per interval as read from the RTC tape. It is created only when MPIF is active and system plots or distributions are requested.
MDWKOUT	Created by the sort program and used as input in JRS3 for MPIF path activity plots and distribution reports.
MSRTIN	Created in JRA2 and passed as input to the sort program by JRA3. This file contains one 381-byte record for each message for the optioned city.
MSRTOUT	Created by the sort program and used as input in JRM4 for the message stream report.
MTRTIN	Created in JRM2 and passed as input to the sort program by JRM2. This file contains one 20-byte record for every message for the optioned cities or LU.
MTRTOUT	Created by the sort program and used as input in JRM4 for terminal activity or LU activity reports (if SNA is generated).
MORTIN	Created in JRA2 and passed as input to the sort program by JRA3. This file contains one 51-byte record for each message on the RTC tape.
MORTOUT	Created by the sort program and used as input to JRM2 for message processing.
DDMMIN	Created in JRA2 and passed as input to the sort program by JRA3. This file contains one 381-byte record for each distributed data management (DDM) message processed by JRA2.
DDMMOUT	Created by the sort program and used as input in JRM5 for the DDM message stream report.

Sort Files Created for Work Space

The following files are created by the sort program and used for work space. See “Sample JCL for Offline Data Reduction” on page 1467 for an example of the definitions of these files.

Note: The space allocated for these work files in the JCL is merely a guideline. See the appropriate sort program books for information about the formulas used to calculate the optimum amount of intermediate work space according to the number of records anticipated.

File	Description
FSRTWK01-FSRTWK06	Work space for sorting FSRTIN.
MARTWK01-MARTWK06	Work space for sorting MARTIN.

File	Description
MDWKWK01	Work space for sorting MDWKIN.
MSRTWK01-MSRTWK06	Work space for sorting MSRTIN.
MTRTWK01-MTRTWK06	Work space for sorting MTRTIN.
MORTWK01-MORTWK06	Work space for sorting MORTIN.
DDMMWK01-DDMMWK06	Work space for sorting DDMMIN.

Intermediate Data Sets

The JRA2 program creates a number of data sets for storing data used in phase 3 of the offline data reduction.

Data Set	Description
HEADER	Contains input for JRA3 (driver report).
STEMP	Contains input for JRS3 (system plot or distribution, or both).
STEMPIS	Contains input for JRS3 (system plot or distribution, or both).
STEMPID	Contains input for JRS3 (idle time plot or distribution, or both).
FILSAV4	Contains input for JRF5 (file plot or distribution, or both).
MPFDF	Contains input for JRM5 (message plot or distribution, or both, by line).
SNPFDF	Contains input for JRM5 (message plot or distribution, or both, by resource ID).

Loosely Coupled Considerations for Running Data Collection and Reduction

One RTC tape file, which can consist of one or more tapes, should be provided for each processor in a loosely coupled complex. Run data collection simultaneously on each TPF image in the loosely coupled complex and then run data reduction for each subsystem in the TPF images.

In a loosely coupled complex, you also need to specify whether you want one specific processor to collect the complexwide data or whether any processor in the TPF system can collect the complexwide data. Specify your preference by using the ZMEAS command.

One specific processor should collect the complexwide data if you want to use the concurrency filter lock facility (CFLF) high-water marks as a tuning aid. ***Do not*** specify a specific processor if there is a possibility of contention for the CFLF locks. If contention occurs for the CFLF locks, check the data reduction reports to determine which processors experienced the contention. However, this information is not available if one specific processor is collecting complexwide data.

Hardware Requirements

For online data collection, the minimum hardware requirements are:
- One tape drive
- TPF online system console.

For offline data reduction, the minimum hardware requirements are:
- MVS system packs
- MVS scratch pack

- One tape drive. Two tape drives are required if the collection data is on 2 tapes.

System Test Compiler

The system test compiler consists of 5 CSECTs that act as 1 MVS program. You can use the system test compiler to do the following:

- Create a pilot tape that contains data records. The pilot tape is loaded to the TPF system using the data loader.
- Create a test unit tape (TUT), which contains application programs, data records, or message records. The TUT tape is used as input to the program test vehicle (PTV).
- Generate standard data and message file (SDMF) tapes. When used in this way, the system test compiler is known as the standard data message update program (SDMU).

See *TPF Program Development Support Reference* for more information about the PTV and the SDMU. See *TPF System Installation Support Reference* for more information about loading a pilot tape.

Input to the System Test Compiler

Input to the system test compiler (or the SDMU) can be loaded to disk. If a tape or a disk is used to load the input, the input is always in card image format.

Input Files

TAPE0
: Standard data and message file (SDMF)

  The SDMF resides permanently on tape. After the SDMF is loaded to a unit test disk, the disk can be set aside and used as SDMF input for subsequent runs.

TAPE3
: Data record information library (DRIL)

  The DRIL normally resides on tape. After the DRIL is loaded to a work disk, the tape can be set aside and used for subsequent runs.

  Note: The data that you set aside can reside on a sequential MVS disk data set.

CARDIN
: Input tape

  If the input to the system test compiler (or the SDMU) is on a tape rather than on control cards, CARDIN must be assigned to the input tape. The input tape contains ***all*** the system test compiler (or the SDMU) control cards and all the control cards required to generate the test unit in card-image format. Make sure that you include the END card.

DISK01
: SDMF/DRIL work disk

  The SDMF/DRIL work disk contains the SDMF and the DRIL. Areas on this disk that are available for the system test compiler are specified on the LOAD ADDRESS cards. This disk can be any supported direct access storage device. (See *TPF Migration Guide: Program Update Tapes* for a list of supported devices.) The SDMU stores any new records that it creates on this disk.

Note: You must use the real-time disk formatter to format the areas of this disk that are available for loading the SDMF and the DRIL. See *TPF Database Reference* for more information about the real-time disk formatter.

STCDD
: Object library data set

  The object library data set contains the system test compiler programs that will be loaded into core. It also contains the application programs that are called by the RUNID test units.

SALTB
: System allocator table

  The system allocator table (SALTBL) contains the information required by the TPF linkage editor to resolve virtual address constants (VCONs). See *TPF System Installation Support Reference* for more information about the SALTBL.

Control Cards

The following information describes the different control cards that are required by the system test compiler and the SMDU.

MVS Job Control Cards (JCL)

The correct sequence of the control cards is as follows, assuming that the system test compiler was linked and stored in a JOBLIB as STC:

```
//STCRUN        JOB      (Accounting Info),MSGLEVEL=1
//JOBLIB        DD       DSN=ACP.LINK.RELvv.ssid,DISP=SHR
//STEPNAME      EXEC     PGM=STC,PARM='SALVER=ss,SALSIZ=nnnnnn'
//SYSOUT        DD       SYSOUT=A
//ISMDD         DD       SYSOUT=A
//LIST          DD       SYSOUT=A
//SYSPRINT      DD       SYSOUT=A
//SYSUDUMP      DD       SYSOUT=A
//STCDD         DD       DSN=ACP.OBJ.RELvv.ssid,DISP=SHR
//SALTB         DD       DSN=ACP.SALTBL.RELvv.ssid,DISP=SHR
//TAPE0         DD       DSN=SDMF,UNIT=(TAPE,,DEFER),LABEL=(,BLP)
//                       DISP=OLD,VOL=SER=userK0
//TAPE1         DD       DSN=NEWTUT,UNIT=(TAPE,,DEFER),LABEL=(1,SL),
//                       DISP=(,PASS),VOL=SER=userK1
//TAPE2         DD       DSN=NEWSDMF,UNIT=(TAPE,,DEFER),LABEL=(,NL),
                         DISP=(,PASS),VOL=SER=userK2
//TAPE3         DD       DSN=DRIL,UNIT=(TAPE,,DEFER),LABEL=(,NL),
//                       DISP=OLD,VOL=SER=userK3
//DISK01        DD       DSN=PARSRCDS,VOL=SER=userd1
//                       UNIT=device,DISP=OLD,
//CARDIN        DD       *
system test compiler control cards
/*
//
```

Where:

vv is the 2-character release or version number.

ssid
is the subsystem ID.

ss is the 2-character alphanumeric version number for the system allocator table (SAL), which is the last 2 characters of the 6-character SAL table member name.

nnnnnn
: is the 1- to 6-digit decimal size of the SAL table.

userxx
: is the correct user volume serial number.

The SDMF and the DRIL can reside on the same tape. In this case, the TAPE3 DD statement must contain the appropriate data set sequence number.

The system test compiler control cards can reside on tape. In this case, the CARDIN DD statement must contain the appropriate information.

When programs are being loaded to the TUT tape, the system test compiler requires that certain information is passed using the PARM= parameter in the EXEC statement. The PARM= parameter provides the version number (SALVER=*vv*) and the size (SALSIZ=*nnnnnn*) of the system allocator table.

Note: If you are creating a TUT tape for use in a subsystem and are including programs from that subsystem, the version of the SAL table must designate the SAL table for the desired subsystem. Programs that do not reside in a subsystem should not be modified for testing in that system.

System Test Compiler Control Cards

The following control cards are necessary for a system test compiler run.

The control cards that define the pilot tape or the TUT tape must be preceded by one or more disk allocation cards and 1 disk type card. The last card of the input must be followed by an END STC card.

Disk Allocation Cards

Each disk allocation card defines a set of disk tracks that were formatted with either 381- or 1055-byte records on the SDMF/DRIL work disk. As many as 20 disk allocation cards for large records and as many as 20 disk allocation cards for small records can be used in one system test compiler run.

Column	Content	Description
1	(blank)	
2–5	LOAD	
6	(blank)	
7–15	ADDRESSES	
16	(blank)	
17–20	xxxx	4-digit hexadecimal address (cylinder number and head number) of the first track of this set.
21	(blank)	
22–23	TO	
24	(blank)	
25–28	yyyy	4-digit hexadecimal address (cylinder number and head number) of the last track of this set.
29	(blank)	
30	Y or N	Specify Y if this set of tracks on the system test compiler work disk was formatted for duplicate records (primary or secondary).

Column	Content	Description
31	(blank)	
32–35	1055 or 381	Size of records on this set of tracks
36–40	(blank)	
41	1	The disk allocation card that contains the first set of large (1055) addresses must contain a 1 in column 41. This card must be the first large address card on all subsequent runs.
42–80	(blank)	

Disk Type Card

One disk type card must be included to indicate to the system test compiler the type of disk pack available for use as a work pack.

Column	Content	Description
1	(blank)	
2–4	STC	Identify the type of run.
5–9	(blank)	
10–13	LOAD	
14–15	(blank)	
16–23	DISKFILE	
24–35	(blank)	
36–39	dddd	Identify the device type of the system test compiler unit test disk; for example, 3390. (See *TPF Migration Guide: Program Update Tapes* for a list of supported devices.)
40	(blank)	
41	L or (blank)	This column must contain an L if the SDMF and the DRIL are loaded to the system test compiler unit test disk from a tape. A blank column indicates to the system test compiler that the SDMF and the DRIL already reside on the system test compiler unit test disk.
42–80	(blank)	

END STC Card

This card must follow the last test unit. It indicates the end of the run.

Column	Content	Description
1	(blank)	
2–4	END	
5	(blank)	
6–8	STC	Identify the type of run.
9–80	(blank)	

SDMU Control Cards

The following control cards are necessary for an SDMU run.

Disk Allocation Cards

The disk allocation cards that are necessary for an SDMU run have the same format as the disk allocation cards that are used for a system test compiler run.

Disk Type Card

One disk type card must be included to indicate to the system test compiler the type of disk pack available for use as a work pack.

Column	Content	Description
1	(blank)	
2–5	SDMU	
6–9	(blank)	
10–13	LOAD	
14–15	(blank)	
16–23	DISKFILE	
24–35	(blank)	
36–39	dddd	Identify the device type of the system test compiler unit test disk; for example, 3390. (See *TPF Migration Guide: Program Update Tapes* for a list of supported devices.)
40	(blank)	
41	L or (blank)	This column must contain an L if the SDMF and the DRIL are loaded to the system test compiler unit test disk from a tape. A blank column indicates to the SMDU that the SDMF and the DRIL already reside on the system test compiler unit test disk.
42–80	(blank)	

SDMU END Card

The SDMU END card follows the SDMU generation cards. It creates a new SDMF tape and ends the job. Any new records that were added to the SDMF are dumped to the printer.

Column	Content	Description
1	(blank)	
2–5	SDMU	Identify type of run.
6–9	(blank)	
10–12	END	
13–80		

System Test Compiler Procedures

1. Mount and make ready all the necessary tapes and disks.
2. Install RUNID decks in the control cards immediately after the load disk file card and in front of the END STC card.
3. Install the system test compiler control cards in the CARDIN device and make ready the CARDIN device.

Note: If you want to load the SDMF and the DRIL on the disk, you must install the load diskfile card. This card must have an L in column 41. Once the SDMF and the DRIL are loaded, you do not have to load them again unless they are changed.

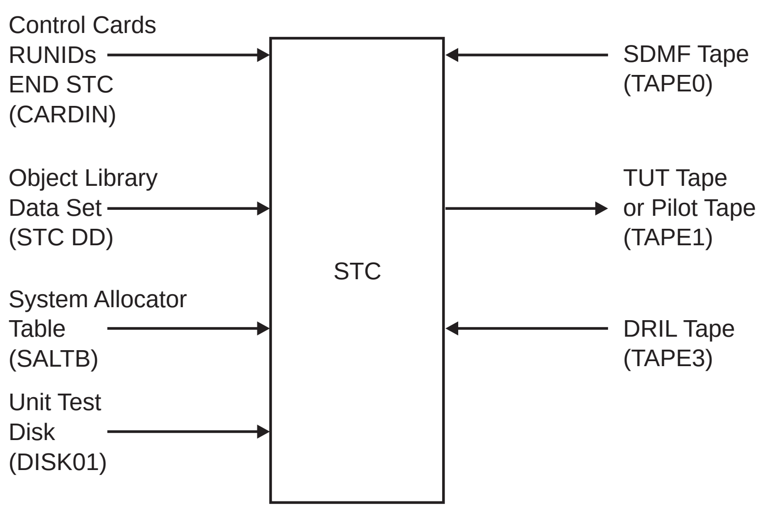
The job card statement is printed on the console and the system test compiler program is loaded.

4. After the system test compiler program and files are loaded, the system test compiler control cards are read from the CARDIN device and the compiled data is generated on the system test compiler output tape. A summary listing is also generated on the SYSPRINT device.

   When the END STC card is read, an end-of-job message is displayed on the console.

   Note: If the multiple database function (MDBF) of the High Performance Option feature is installed, you must use an SDMF/DRIL tape and system test compiler program that is generated for the subsystem where the pilot tape or TUT tape is being created.

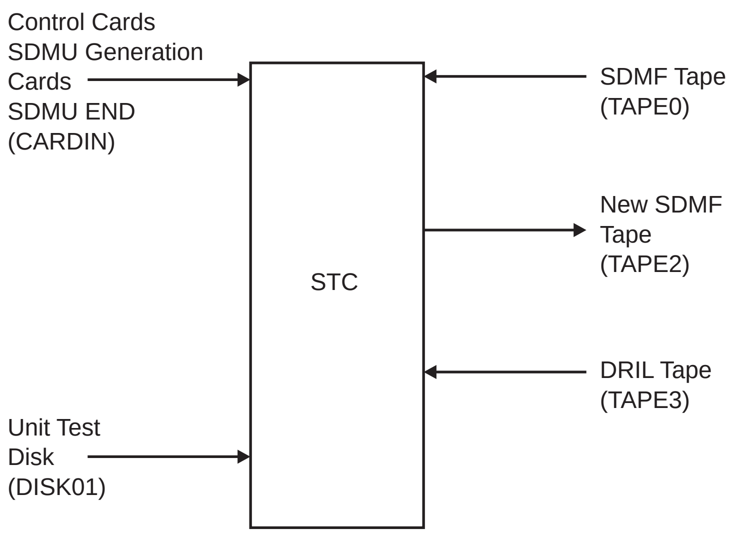
System Test Compiler Run Setup



If a tape is used as input, the system test compiler control cards (beginning with the LOAD ADDRESS cards), the RUNID units, and the END STC card are on this tape, and it is assigned symbolically as CARDIN.

Note: If the SDMF and the DRIL already reside on the SDMF/DRIL work disk, you do not need to mount the SDMF/DRIL tapes.

SDMU Run Setup



If a tape is used as input, the system test compiler control cards (beginning with the LOAD ADDRESS cards), the SDMU generation cards, and the SDMU END card are on this tape, and it is assigned symbolically as CARDIN.

Note: If the SDMF and the DRIL already reside on the SDMF/DRIL work disk, you do not need to mount the SDMF/DRIL tapes.

Output from the System Test Compiler

The following information describes the output produced by the system test compiler.

TUT Tape or Pilot Tape (TAPE1)

When the system test compiler is run, the output tape contains all of the pilot systems or test units created during that run. If this is a pilot tape, the tape is used as input for the data loader (see *TPF System Installation Support Reference*). If this is a TUT tape, the tape is used as input for the PTV. See *TPF Program Development Support Reference* for more information about the TUT tape and the PTV.

SDMF Tape (TAPE2)

When the SDMU is run, a new SDMF tape is created. See the SDFPF data macro for more information about the SDMF tape.

Summary Listing

All the cards read during the run are displayed on a listing that is generated on the SYSPRINT device. If any errors occurred, the error messages are also displayed on the listing.

Normal Messages

The JOB card is printed by the MVS system at the beginning of the run.

An EOJ message is printed by the system test compiler at the end of the system test compiler (or the SDMU) run.

Error Messages

There are no system test compiler (or SMDU) error messages that require operator intervention in order to continue the job. If an error message is displayed on the console, the job either continues or ends as explained in the message. See *Messages (System Error and Offline)* and *Messages (Online)* for more information about the error messages that are displayed.

Hardware Requirements

The following configuration is required to run the system test compiler (or the SDMU):

- An MVS system with at least 512KB of main storage
- One MVS-supported 80-column reader device
- One MVS-supported 132-character printer device
- Two IBM-supported disk drives (see *TPF Migration Guide: Program Update Tapes* for a list of supported devices)
- Three IBM-supported 3400-series tape drives.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

Q

R

S

T

U

V

W

X

Y

Z

File Number: S370/30XX-40
Program Number: 5748-T14

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SH31-0162-15