

Transaction Processing Facility



# System Generation

*Version 4 Release 1*



Transaction Processing Facility



# System Generation

*Version 4 Release 1*

**Note!**

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

**Sixteenth Edition (June 2002)**

This is a major revision of, and obsoletes, SH31-0171-14 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.

© **Copyright International Business Machines Corporation 1994, 2002. All rights reserved.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>Figures</b> . . . . .	ix
<b>Tables</b> . . . . .	xi
<b>Notices</b> . . . . .	xiii
Trademarks . . . . .	xiii
<b>About This Book</b> . . . . .	xv
Who Should Read This Book . . . . .	xv
Conventions Used in the TPF Library . . . . .	xv
How to Read the Syntax Diagrams . . . . .	xvi
Related Information . . . . .	xx
IBM Transaction Processing Facility (TPF) 4.1 Books . . . . .	xx
IBM High-Level Language Books . . . . .	xx
Miscellaneous IBM Books . . . . .	xx
Online Information . . . . .	xxi
How to Send Your Comments . . . . .	xxi

---

## Part 1. Planning the TPF System. . . . . 1

<b>Introduction to System Generation</b> . . . . .	3
System Design . . . . .	5
TPF Capabilities . . . . .	7
Tightly Coupled Processor Support . . . . .	8
TPF Communication Support . . . . .	8
High Performance Option . . . . .	9
Multiple Database Function . . . . .	9
Loosely Coupled Complexes . . . . .	10
Multi-Processor Interconnect Facility . . . . .	11
Interrelationship of Multiprocessing Components . . . . .	12
<b>System Generation Process</b> . . . . .	15
System Initialization Overview . . . . .	15
System Initialization Package . . . . .	16
Non-SNA Communication Keypoint Records . . . . .	16
SNA Communication Tables . . . . .	16
C Language Support . . . . .	16
TPF System Loading . . . . .	17
Program/Keypoint Loading . . . . .	17
Loader General File . . . . .	18
System Restart . . . . .	20
Database Loading . . . . .	22
Communication Device Loading . . . . .	26
Post-Cutover Activities . . . . .	26

---

## Part 2. Defining the TPF System . . . . . 29

<b>Processor and Main Storage Resources</b> . . . . .	31
Processor Definition . . . . .	31
TPF Main Storage Layout . . . . .	33
TPF Control Program Area . . . . .	34
Application Program Areas . . . . .	36

Application Global Area . . . . .	40
SIP for Globals . . . . .	40
Control Program Records and Table Area . . . . .	41
Working Storage . . . . .	41
Storage Requirements for Pool Directory Buffers . . . . .	44
Storage Requirements for Multi-Processor Interconnect Facility . . . . .	45
Storage Required For Tables . . . . .	45
Storage Required For Control Blocks . . . . .	46
Storage Required for Staging Buffers . . . . .	47
Sample Case for Storage Requirements . . . . .	47
Tightly Coupled Scheduler . . . . .	48
Function . . . . .	48
Tuning the Tightly Coupled Scheduler . . . . .	48
Scheduler Algorithm . . . . .	50
Tape Support . . . . .	52
Planning Your Tape Configuration . . . . .	52
Configuring Devices . . . . .	53
Customizing Tape Processing Options . . . . .	53
Customizing Stalled Tape Module Queue Monitoring . . . . .	54
Allocating Additional Working Storage for LIFO Devices . . . . .	54
Defining Default Categories for the Tape Library Dataserver . . . . .	54
Defining a Minimum Tape Level for the Tape Library Dataserver . . . . .	55
Unit Record Support . . . . .	55
<b>Database Support . . . . .</b>	<b>57</b>
Disk File Storage . . . . .	61
Fixed File Storage . . . . .	63
RAMFIL Statements for Fixed Record Types . . . . .	65
Pool File Storage . . . . .	68
Ratio Dispensing . . . . .	70
Fallback . . . . .	71
Keypoint Records . . . . .	73
General Data Sets . . . . .	77
General Files . . . . .	78
Database Device Capacity . . . . .	79
Database Device Addressing . . . . .	79
Alternate Paths to Disk Storage . . . . .	87
Permanent Record Area . . . . .	88
Database Allocation . . . . .	91
Record Types and Space Requirements . . . . .	91
UFT/FTI Examples . . . . .	93
Record Duplication . . . . .	98
Address Assignment . . . . .	99
File Design Worksheets . . . . .	99
Partially Duplicated Layout Example . . . . .	100
System Utilized Record Types . . . . .	103
Standard Data/Message File . . . . .	121
<b>Data Communications Support . . . . .</b>	<b>123</b>
Basic TPF Communications Support . . . . .	123
High Performance Option Feature . . . . .	124
Loosely Coupled Communications Considerations . . . . .	124
Multiple Database Function Communications Considerations . . . . .	125
Communications Record Generation . . . . .	125
SIP Communication Macros . . . . .	126
Communication Keypoint Macros (Non-SNA) . . . . .	132

Communication Pilot Record Generation . . . . .	137
SNA Table Generation . . . . .	139
3600/4700 SNA Generation Requirements . . . . .	140
<b>Application and System Support . . . . .</b>	<b>141</b>
Application Program Interface . . . . .	141
Message Router Support . . . . .	142
Message Router Application Attributes . . . . .	143
Message Router Support in an ACF Network . . . . .	143
Message Router Support for the High Performance Option Feature . . . . .	144
Message Router Implementation Guidelines . . . . .	144
Operator, Terminal, and LU Identification and Control . . . . .	145
Log Processor . . . . .	145
Operator/Terminal Control . . . . .	146
Communications Source Message Router and System Message Processor . . . . .	148
UI/Reservation Package (PARS) . . . . .	149
Unsolicited Message Support . . . . .	150
Long Message Transmitter Package . . . . .	151
Message Switching Packages . . . . .	152
Domestic Message Switching . . . . .	152
World Trade Message Switching . . . . .	152
TPF Mapping Support . . . . .	152
Control Program User Exits . . . . .	152
Global Area . . . . .	153
Data Collection and Reduction . . . . .	153
Database Reorganization (DBR) . . . . .	154
Recoup . . . . .	154
File Capture and Restore . . . . .	155
Loaders . . . . .	155
TPF General File Loader . . . . .	155
E-Type Loader . . . . .	156
Auxiliary Loader . . . . .	156
Data Loader . . . . .	156
Pool Directory Generation and Maintenance . . . . .	157
Program Test Vehicle . . . . .	157
System Test Compiler . . . . .	158
Diagnostic Output Formatter . . . . .	159
Real-Time Disk Formatter . . . . .	161
FACE Driver and Offline Interface . . . . .	162
System Clocks . . . . .	162
TPF Collection Support . . . . .	162

---

## Part 3. Generating the TPF System . . . . . 165

<b>Introduction to SIP . . . . .</b>	<b>167</b>
Stage I: Producing the Job Stream . . . . .	167
Stage II: Processing the Job Stream . . . . .	167
SIP Functions . . . . .	169
Stage I . . . . .	170
Stage II . . . . .	170
Before SIP . . . . .	171
Unpacking . . . . .	171
System Requirements . . . . .	172
Preparing for Stage I . . . . .	175
Stage II Job Card . . . . .	175
SYSPUNCH Output . . . . .	175

Creating Program Tables . . . . .	178
WTC Support . . . . .	183
Creating a Site System Allocator . . . . .	183
Description of the Allocator Decks . . . . .	183
Record ID Attribute Table (RIAT) . . . . .	184
Creating the RIAT . . . . .	184
Accessing the RIAT Online . . . . .	185
FACE Table Generation . . . . .	186
FACE Table Generator Input . . . . .	186
FACE Table Generator Output . . . . .	186
Building a FACE Table . . . . .	187
Compiling the FACE Table Generator . . . . .	188
FACE Table Generator Parameters . . . . .	194
Running the FACE Table Generator . . . . .	195
FACE Table Generator Error Messages . . . . .	198
FACE Table Generator Report Listing . . . . .	198
Assembling CONK, CTSD, and IRCCR . . . . .	205
<b>Stage I: Coding the SIP Macros . . . . .</b>	<b>207</b>
Coding TPF with the Multiple Database Function . . . . .	207
BBSAT . . . . .	208
BSNCT . . . . .	210
CCPERR . . . . .	212
CCPPOL . . . . .	214
CCPSTA . . . . .	216
CLOCKS . . . . .	217
CONFIG . . . . .	219
CORREQ . . . . .	229
CRASTB . . . . .	234
DATACO . . . . .	240
DDCCAP . . . . .	242
GENFIL . . . . .	245
GENSIP . . . . .	247
GLOBAL . . . . .	255
GLSYNC . . . . .	258
INDSN . . . . .	261
IODEV . . . . .	264
LINES . . . . .	267
LOGCAP . . . . .	269
MSGRT . . . . .	271
MSGRTA . . . . .	272
NETWK . . . . .	279
ONLFIL . . . . .	282
RAM . . . . .	286
RAMEND . . . . .	289
RAMFIL . . . . .	290
RESCAP . . . . .	302
RIATA FINISH . . . . .	304
RIATA ID . . . . .	305
RIATA START . . . . .	311
SSDEF . . . . .	312
SYNCLK . . . . .	314
UFTEND . . . . .	317
UFTFTI . . . . .	318
UNITRD . . . . .	321
USEREQ . . . . .	322



UTPROT . . . . .	326
<b>Stage II: Executing the Job Stream . . . . .</b>	<b>329</b>
Analyzing Errors . . . . .	329
SIP Report . . . . .	329
Description of Optional Support . . . . .	329
Detailed Description of User Macros . . . . .	329
Revisions to Job Stream . . . . .	329
Types of Output . . . . .	330
Post-SIP Stage II . . . . .	330
Other Generation Processes Executed During Stage II. . . . .	330
GTSZ Getsize. . . . .	330

---

<b>Part 4. Generating the TPF Communications Network. . . . .</b>	<b>333</b>
---	------------

<b>Non-SNA Communications . . . . .</b>	<b>335</b>
SCK Generation Preparation . . . . .	335
SCK Generation Phase I. . . . .	335
SCK Generation Phase II . . . . .	336
Specifying the Communications Network . . . . .	337
SKLNG . . . . .	338
PKSTG . . . . .	342
SENDG . . . . .	343
Creating the Non-SNA Communications Pilot Tape . . . . .	345
Inner Macros . . . . .	345
<b>Appendix A. System Generation Messages . . . . .</b>	<b>349</b>
<b>Appendix B. Sample SIP Input and Report . . . . .</b>	<b>351</b>
Sample SIP Stage I Input . . . . .	351
Sample SIP Stage I Output - SIP Report . . . . .	472
<b>Appendix C. SIP Stage II Job Summary . . . . .</b>	<b>473</b>
<b>Appendix D. Sample Central Site Configurations . . . . .</b>	<b>485</b>
Base System Configuration . . . . .	487
Loosely Coupled System Configuration . . . . .	488
<b>Appendix E. SIP Library References . . . . .</b>	<b>493</b>
ACP.SYMACRO.RELv Members . . . . .	493
ACP.SYSRCE.RELv Members . . . . .	494
<b>Appendix F. SIP Stage II Data Sets . . . . .</b>	<b>495</b>
<b>Index . . . . .</b>	<b>497</b>



---

## Figures

1. Initial vs. Online Restart. . . . .	5
2. TPF Design Process . . . . .	6
3. MPIF Complex . . . . .	12
4. Interrelationship of Multiprocessing Components . . . . .	14
5. TPF Program/Keypoint Loading . . . . .	18
6. Allocation of TPF Online Modules. . . . .	20
7. Loading the TPF Restart Area . . . . .	22
8. Program Maintenance Overview . . . . .	27
9. Virtual Storage Layout . . . . .	35
10. Logical/Physical Disk Device Relationship . . . . .	58
11. Record Allocation. . . . .	59
12. Multiple Disk Device Type Record Organization . . . . .	59
13. File Duplication Methods . . . . .	62
14. Fixed Record Space . . . . .	64
15. Three Dimensional Example of Record Uniqueness . . . . .	67
16. Sample File Pool Directory . . . . .	69
17. Example of a 32-Device Controller . . . . .	80
18. Example of a 16-Device Controller . . . . .	81
19. Example of a 64-Device Controller . . . . .	82
20. Sample SIP Parameters for Online Modules . . . . .	83
21. Module Status Table Layout Example . . . . .	84
22. Symbolic Module Numbers Example . . . . .	84
23. FARF4, FARF5, and FARF6 Addresses . . . . .	94
24. 3380 Device Sample Layout . . . . .	102
25. Sample Device RAMFIL Macros. . . . .	103
26. Functions Provided by SIP . . . . .	169
27. JCL for Allocating TPF Data Sets . . . . .	177
28. JCL for Allocating TPFDF Data Sets . . . . .	177
29. Sample Configuration. . . . .	228
30. Sample Base-Only System Configuration . . . . .	486
31. Sample Loosely Coupled System Configuration . . . . .	490



---

## Tables

1. Application Program Areas above the 16M Line . . . . .	38
2. Main Storage Structure Sizes . . . . .	43
3. Database Areas . . . . .	61
4. DEVA Pool Section Assignments . . . . .	70
5. Control Program Keypoints . . . . .	75
6. General File Worksheet . . . . .	78
7. Direct Access Storage Device Capacities . . . . .	79
8. Direct Access Device Worksheet . . . . .	85
9. Disk Online Module List Example . . . . .	86
10. Volume Serial Number Worksheet . . . . .	86
11. Permanent Record Area Layout of the Restart Area . . . . .	88
12. Online Restart Area Work Sheet (BSS or BSS in an MDBF System) . . . . .	90
13. Fixed File Data Record Worksheet . . . . .	92
14. Module-to-Module Duplication Worksheet . . . . .	99
15. Four-Module 3380 Sample File Layout . . . . .	100
16. System-Required Fixed File Record Types . . . . .	105
17. Global Area Record Types . . . . .	110
18. Records Checked by the FACE Table Generator (FCTBG) . . . . .	111
19. Records Checked during SIP Stage I Assembly . . . . .	113
20. SIP Communications Macros . . . . .	126
21. Non-SNA Transmission Control Unit (TCU) Worksheet . . . . .	134
22. Non-SNA Communication Lines Worksheet. . . . .	136
23. RAMFIL Parameter Rules . . . . .	300
24. Required Output Data Sets for SIP Stage II . . . . .	495



---

## 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:

- AD/Cycle
- APPN
- Advanced Peer-to-Peer Networking
- C/370
- C++/MVS
- CICS
- DB2
- Enterprise Systems Connection Architecture
- EOCF/2

- ESCON
- ES/3090
- ES/9000
- IBM
- MQSeries
- MVS/ESA
- MVS/SP
- OS/390
- Sysplex Timer
- System/360
- System/370
- System/390
- VisualAge
- VTAM
- 3090.

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



---

## About This Book

Use this book to plan for and install a TPF system. To migrate to a new release of the TPF system, use this book with *TPF Migration Guide: Program Update Tapes*. It includes detailed descriptions on the use of the system initialization program (SIP) for generating the system and the facilities for generating the non-SNA communications network.

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.

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.

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*.

See “Communicating Your Comments to IBM” for more information about how to contact your TPF support representative.

---

## Who Should Read This Book

This book is written for system programmers responsible for the planning and installation of a TPF system. It should be used with *TPF System Installation Support Reference*, which describes closely related functions and tools that support the system generation process.

Users of this book should have a basic understanding of the TPF system, and the communication facilities supported by the TPF system. Familiarity with MVS systems is also required. For a comprehensive technical overview of the TPF system, see *TPF Concepts and Structures*.

---

## Conventions Used in the TPF Library

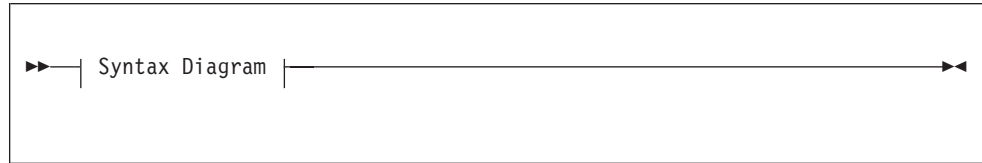
The TPF library uses the following conventions:

Conventions	Examples of Usage
<i>italic</i>	<p>Used for important words and phrases. For example:  <i>A database</i> is a collection of data.</p> <p>Used to represent variable information. For example:  Enter <b>ZFRST STATUS MODULE</b> <i>mod</i>, where <i>mod</i> is the module for which you want status.</p>
<b>bold</b>	<p>Used to represent text that you type. For example:  Enter <b>ZNALS HELP</b> to obtain help information for the ZNALS command.</p> <p>Used to represent variable information in C language. For example:  <b>level</b></p>
monospaced	<p>Used for messages and information that displays on a screen. For example:  PROCESSING COMPLETED</p> <p>Used for C language functions. For example:  maskc</p> <p>Used for examples. For example:  maskc(MASKC_ENABLE, MASKC_IO);</p>
<b><i>bold italic</i></b>	<p>Used for emphasis. For example:  You <b><i>must</i></b> type this command exactly as shown.</p>
<b><u>Bold underscore</u></b>	<p>Used to indicate the default in a list of options. For example:  <b>Keyword=OPTION1   <u>DEFAULT</u></b></p>
Vertical bar	<p>Used to separate options in a list. (Also referred to as the OR symbol.) For example:  <b>Keyword=Option1   Option2</b></p> <p><b>Note:</b> Sometimes the vertical bar is used as a <i>pipe</i> (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.</p>
CAPital LETters	<p>Used to indicate valid abbreviations for keywords. For example:  KEYWord=<i>option</i></p>
Scale	<p>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.</p> <p> ...+...1...+...2...+...3...+...4...+...5...+...6...+...7...</p> <p>LOADER    IMAGE   CLEAR</p> <p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. The word LOADER must begin in column 1.</li> <li>2. The word IMAGE must begin in column 10.</li> <li>3. The word CLEAR must begin in column 16.</li> </ol>

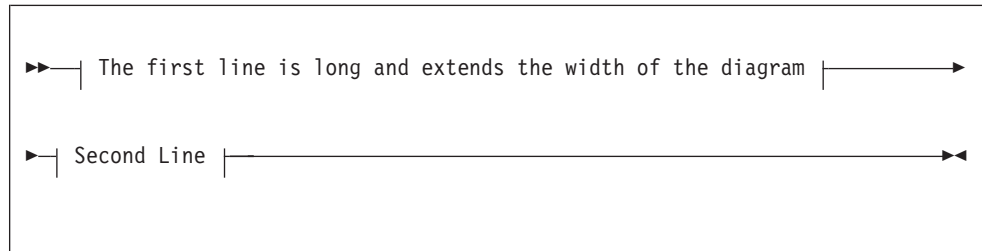
## How to Read the Syntax Diagrams

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

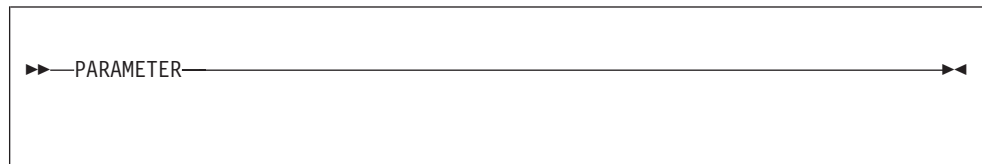
- Read the diagrams from left-to-right, top-to-bottom, following the main path line. Each diagram begins on the left with double arrowheads and ends on the right with 2 arrowheads facing each other.



- 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.

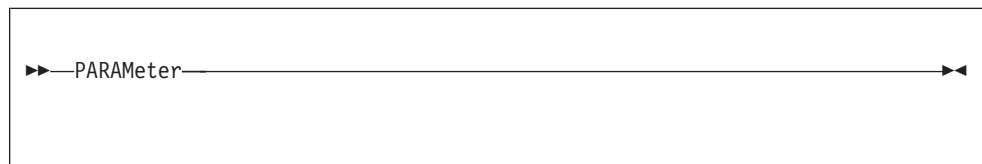


- A word in all uppercase is a parameter that you must spell ***exactly*** as shown.

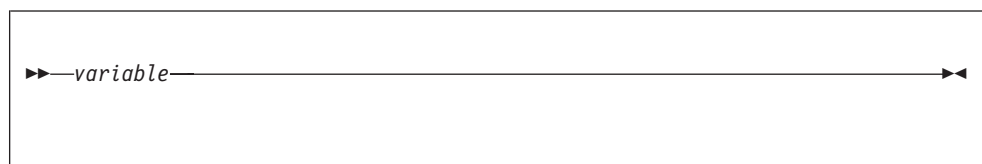


- If you can abbreviate a parameter, the optional part of the parameter is shown in lowercase. (You must type the text that is shown in uppercase. You can type none, one, or more of the letters that are shown in lowercase.)

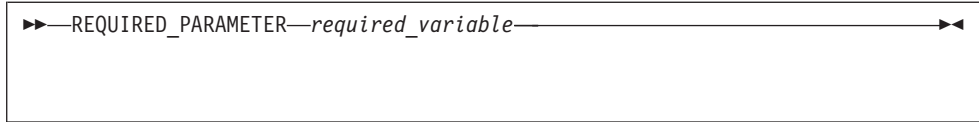
**Note:** Some TPF commands are case-sensitive and contain parameters that must be entered exactly as shown. This information is noted in the description of the appropriate commands.



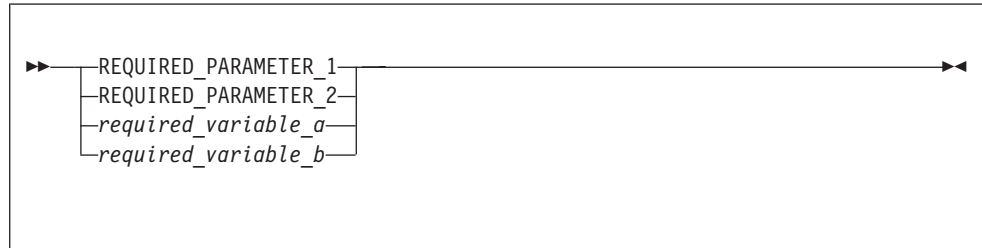
- A word in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.



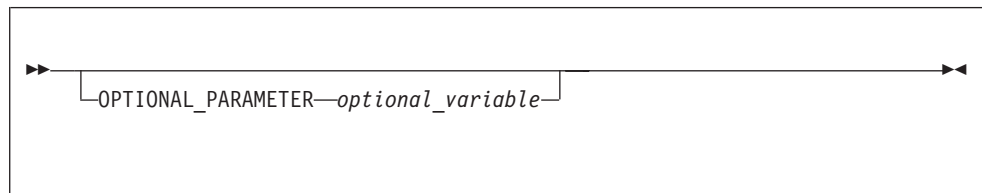
- Required parameters and variables are shown on the main path line. You must code required parameters and variables.



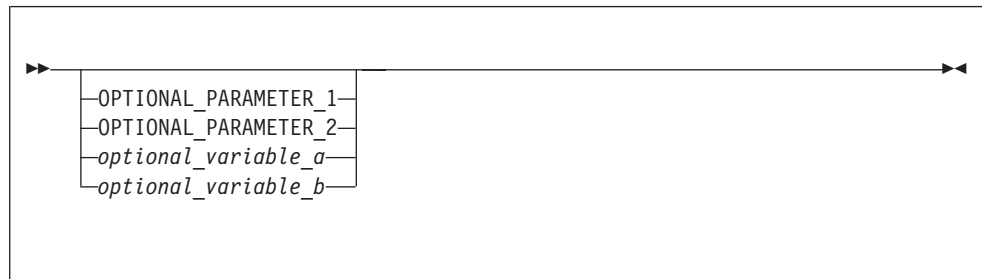
- If there is more than one mutually exclusive required parameter or variable to choose from, they are stacked vertically.



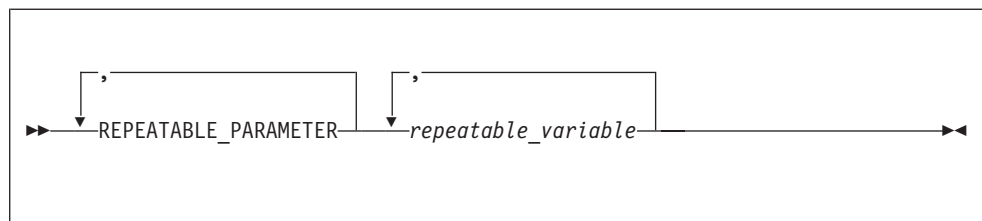
- Optional parameters and variables are shown below the main path line. You can choose not to code optional parameters and variables.



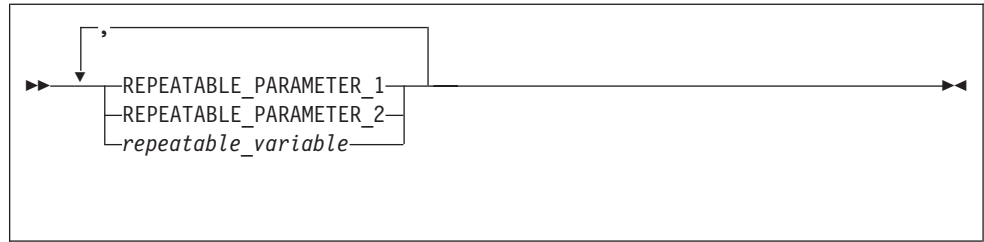
- If there is more than one mutually exclusive optional parameter or variable to choose from, they are stacked vertically below the main path line.



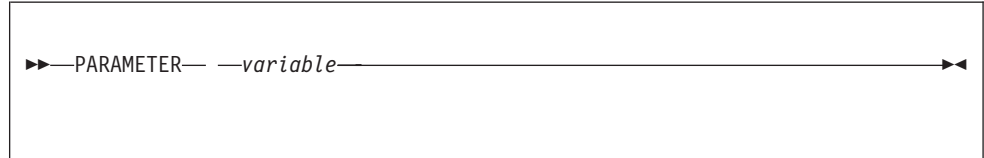
- An arrow returning to the left above a parameter or variable on the main path line means that the parameter or variable can be repeated. The comma (,) means that each parameter or variable must be separated from the next parameter or variable by a comma.



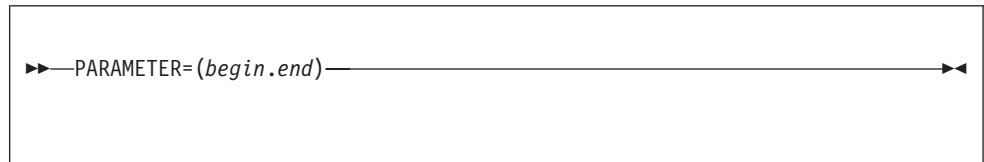
- An arrow returning to the left above a group of parameters or variables means that more than one can be selected, or a single one can be repeated.



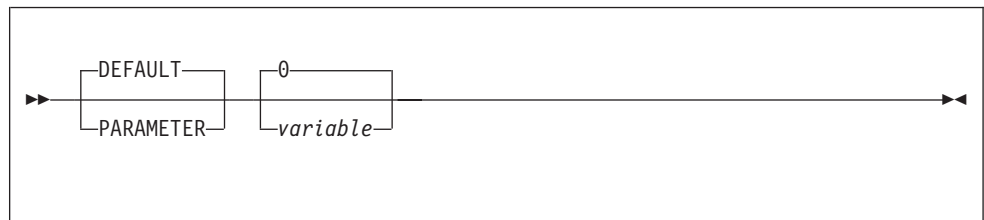
- If a diagram shows a blank space, you must code the blank space as part of the syntax. In the following example, you must code **PARAMETER** *variable*.



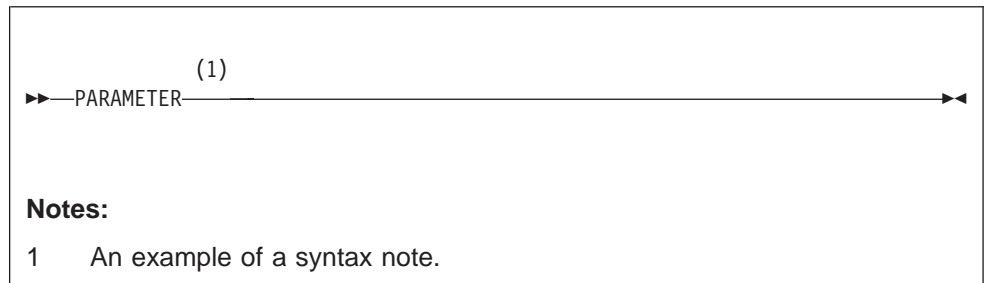
- If a diagram shows a character that is not alphanumeric (such as commas, parentheses, periods, and equal signs), you must code the character as part of the syntax. In the following example, you must code **PARAMETER=(begin.end)**.



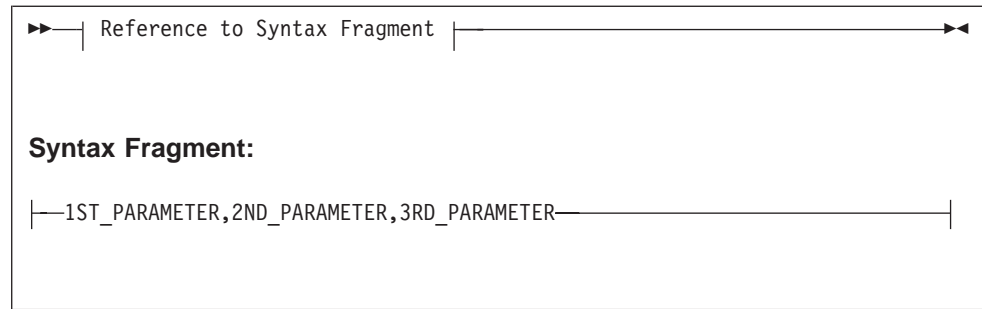
- Default parameters and values are shown above the main path line. The TPF system uses the default if you omit the parameter or value entirely.



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



- 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.



---

## 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 Application Programming*, SH31-0132
- *TPF ACF/SNA Network Generation*, SH31-0131
- *TPF ACF/SNA Data Communications Reference*, SH31-0168
- *TPF System Installation Support Reference*, SH31-0149
- *TPF Main Supervisor Reference*, SH31-0159
- *TPF Operations*, SH31-0162
- *TPF Database Reference*, SH31-0143
- *TPF Data Communications Services Reference*, SH31-0145
- *TPF System Performance and Measurement Reference*, SH31-0170
- *TPF Migration Guide: Program Update Tapes*, GH31-0187
- *TPF General Macros*, SH31-0152
- *TPF Concepts and Structures*, GH31-0139
- *TPF Program Development Support Reference*, SH31-0164
- *TPF Non-SNA Data Communications Reference*, SH31-0161
- *TPF C/C++ Language Support User's Guide*, SH31-0121
- *TPF System Macros*, SH31-0151
- *TPF Transmission Control Protocol/Internet Protocol*, SH31-0120.

### IBM High-Level Language Books

- *IBM C/C++ for MVS/ESA C++/MVS Language Reference*, SC09-1992
- *OS/390 C/C++ User's Guide*, SC09-2361.

### Miscellaneous IBM Books

- *ES/9000, ES/3090 Input/Output Configuration Program User's Guide and ESCON Channel-to-Channel Reference*, GC38-0097
- *IBM System/370 Input/Output Configurator*, GA22-7002
- *3704/3705 Control Program Generation Utilities Guide*, GC30-3008
- *General Information - Binary Synchronous Communications*, GA27-3004
- *IBM Extended Operations Console Facility/2 System Administrator's Guide*, SH31-0105
- *3600 Finance Communication System 3614 Programmer's Guide and Reference*, GC27-0010

- *IBM 3600 Finance Communication System Instructions and Macro Reference*, GC27-0003
- *IBM 3600 Finance Communication System Programmer's Guide and Component Descriptions*, GC22-9045
- *ESA/390 Principles of Operation*, SA22-7201
- *DSF User's Guide and Reference*, GC35-0033
- *MVS/ESA JCL Reference*, (order the correct version and release for your installation)
- *EP for 3705 Generation and Utilities Guide*, SC30-3242
- *OS/VS2 MVS Utilities*, GC26-3902
- *3990 Transaction Processing Facility Support RPQs*, GA32-0134.

## Online Information

- *Messages (Online)*
- *Messages (System Error and Offline)*.

---

## 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](mailto: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. Planning the TPF System



---

# Introduction to System Generation

An operational system is a combination of the TPF system, application programs, and people. People assign purpose to the system and use the system. The making of an operational system depends on three interrelated concepts:

- System definition: The necessary application and TPF system knowledge required to select the hardware configuration and related values used by the TPF system software.
- System initialization: The process of creating the TPF system tables and configuration-dependent system software.
- System restart and switchover: The procedures used by the TPF system software to ready the configuration for online use.

The first two items are sometimes collectively called *system generation*; also *installing* and *implementing*. System definition is sometimes called *design*. System restart is the component that uses the results of a system generation to place the system in a condition to process real-time input. The initial startup is a special case of restart and for this reason system restart is sometimes called *initial program load*, or IPL. System restart uses values found in tables set up during system generation and changed during the online execution of the system. A switchover implies shifting the processing load to a different central processing complex (CPC), and requires some additional procedures on the part of a system operator. A restart or switchover may be necessary either for a detected hardware failure, detected software failure, or operator option. In any event, system definition (design), initialization, restart, and switchover are related to error recovery. This provides the necessary background to use this publication, which is the principal reference to be used to install the TPF system.

Performing a system generation requires a knowledge of the TPF system structure, system tables, and system conventions, a knowledge of the applications that will be programmed to run under the system, and a user's knowledge of multiple virtual storage (MVS). The TPF system and the application knowledge are required to make intelligent decisions to accomplish the system definition of a unique TPF system environment. The use of MVS is necessary because many programs used to perform a system generation run under control of MVS. Although this publication does not rely on much MVS knowledge, when the moment arrives to use the implementation manuals, the necessary MVS knowledge must be acquired. You are assumed to have some knowledge of the S/370 assembly program as well as jargon associated with the MVS operating system. Some knowledge of C language is also helpful, because some of the programs that are used to generate the system are written in C.

System definition is a nontrivial endeavor because both TPF system and application knowledge is necessary. For example, the frequency of use of an application program should be considered in making the decision for assigning the program to system residence; for example, main or file storage. An understanding of the basic data organization of the system is necessary to allocate the appropriate area of online file storage for *all* the programs. All TPF system programs and application programs that are used in the online environment are placed in file storage. Some programs are permanently assigned to main storage to control the online environment or to improve the execution efficiency of frequently used ECB-controlled programs.

An appreciation of defining a system is necessary to relate a unique TPF system to its environment. Clearly, no amount of documentation can anticipate all the variants of applications to be run under the system. However, the interrelationship of the applications, the TPF control program, and the corresponding data structures is of paramount importance. The initial system definition must be approached with the best available information. Usually, even this is not sufficient for an accurate definition. Through knowledge, analysis, and an understanding of system guidelines, a workable operational system can be defined and initialized. A system definition is not as insurmountable as it may first seem, because the TPF system includes components that allow the system programmer to change the system tables and programs, the application programs, and the physical components. In this sense, the operational system can be adapted to its environment.

The initial system generation, which requires analysis techniques and knowledge of the TPF system, represents the beginning of a continuous process required throughout the life of an operational system.

The terms *system initialization* and *initialization process* are used to emphasize the human activity and *offline* procedures required to produce programs and data to be placed on *online* system resident storage. The term *offline procedures* refers to the execution of programs, written to run under MVS, that assist in the production of the online programs and data. (The term *procedures* is generally used to mean a program or collection of programs.) The goal of system initialization is to place the various levels of TPF system storage facilities into the condition where restart procedures may be invoked to allow online processing to begin. The restart procedures are involved with the programs that build system records used to control online processing. Many of these records are assigned values identified by the initialization process. Some important distinctions of terminology follow:

- A program in the restart procedures, called the initializer program (the TPF system identification is CCCTIN), uses values assigned to system records used for main (core) storage management. The *initializer program* is not the same thing as the *initialization process*. The initializer program is a portion of the restart procedures.
- A system restart begins by pressing the initial program load (IPL) key. This invokes a standard S/390 IPL sequence, which in turn invokes the TPF system IPL program. The IPL program loads enough of the system to main storage to permit the initializer program to allocate system tables to main storage. In a loosely coupled complex, an IPL requires the coordination of each of the CPCs in the complex; if a CPC is a multiprocessor, the IPL is performed by only one of the CPUs.

An *initial* restart differs from an *online* restart, which is shown in Figure 1. An initial restart uses input data and restart programs created in the offline environment. An online restart uses some of the same data, but the data and restart programs are accessed from online files. The basic restart procedure of loading the TPF core resident programs and executing the initializer program is identical for both the initial restart or the online restart. The location of the restart programs and data identifies the difference. During an initial restart, additional programs to load programs and data to system storage must be invoked. Normally, much of the data put in place during an initial restart does not need to be reloaded during an online restart. (If a reload is necessary, an initial restart is required.) The basic restart programs (such as the IPL and initializer programs) and the structure of the data that these restart programs process are identical. The location of the data is different, as well as some of the data content.

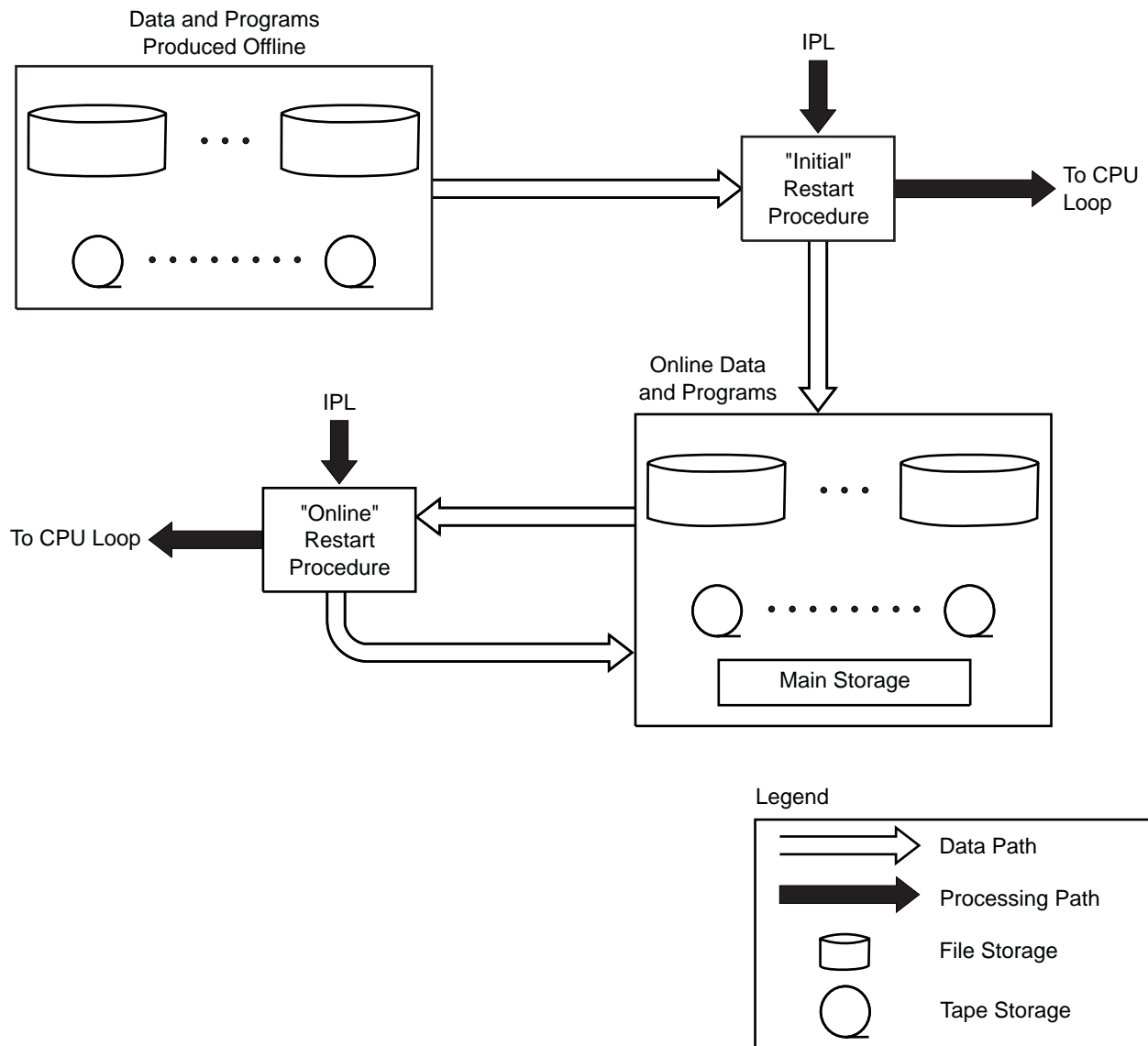


Figure 1. Initial vs. Online Restart

## System Design

Certain design activities must be completed before beginning the system generation procedures described in this publication. These activities include:

- Hardware configuration and main storage configuration
- Database design
- Communications network design
- Application program design.

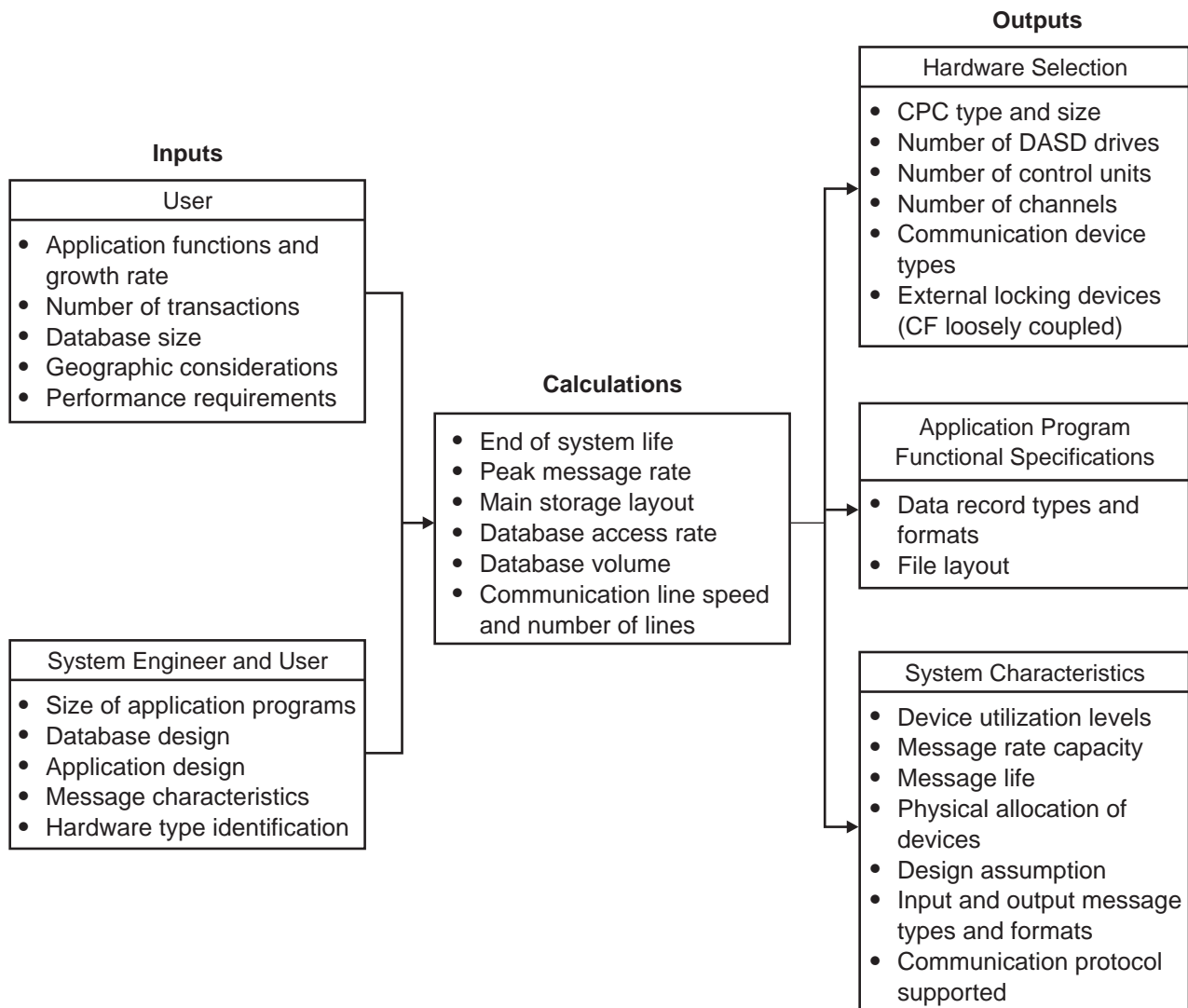


Figure 2. TPF Design Process

In standard IBM operating systems the substitution of real values for symbolic names (such as real addresses for symbolic labels) is postponed as long as possible to maintain flexibility for the programmers and the operators. An example is the process of locating a data set for use by an application program. Under MVS control, this data set may be defined at the time the application program is compiled (in the DCB), at JCL time (in the job control language), or during execution of the program (from the data set label during the open). While postponing substitution does increase flexibility, there is usually a penalty in terms of system performance.

In the TPF system, much of the substitution is done during system generation. By substituting during system generation, the system designer (and not the application programmer or the operations supervisor) makes the decisions that affect system performance. The design of TPF, with its heavy emphasis on performance, dictates that system resources should not be used for substitution that can be done by a thorough job of system design and planning.

Design decisions such as data set placement, database definition, and access method requirements are made during the system design phase. Front loading of

the system design by TPF helps ensure that the system architects actually design the system and that errors during operation or application development will not affect the performance of the system. The entire TPF file structure must be designed before generating a TPF system. Data set placement, record sizes, organization, and file types must be decided during system design. This information is used during system generation to build internal tables that provide direct access to all programs and data records.

A TPF designer should allow for as much future growth as is feasible by specifying more at generation time than is required. This will compensate for the emphasis TPF places on system generation and allow additions to be made to the system as needed. Even so, changes in the file layout, communications network, or other areas may require the rerunning of steps in the generation process. System generation is usually not done just once in a TPF system. Figure 2 presents an overview of the design process.

---

## TPF Capabilities

The requirements of your TPF system determine which features to include. In some cases certain functions are mandatory, while others are optionally selected by the user. Unique inputs or variables must be provided for the mandatory support (such as hardware addresses). However, the optional software support required for the system being installed is even more important (for example, unit record support). A significant consideration is the interdependencies of the software. If, for example, 3270/SDLC support is required, systems network architecture (SNA) support is mandatory. Once interdependencies are sorted out, the various options in a given area may be chosen.

One approach to this problem is to review this publication, decide on the options needed, and then go through the publication again to determine how to actually generate the system. Another approach, which can be combined with the first, is to break down the system into its major components: database support, communications support, and applications support. A problem with this approach is that there are many interrelationships between these areas (for instance, to determine the DASD requirements, the file resident record requirements of the communications protocol being used must be known.)

The mandatory portion of the basic system includes the fixed and pool record support. While fixed and pool file records must be stored on DASD, users select the DASD model. Some fixed and pool records are required by the system and some are required by the optional software packages. In addition, users must define the application database, including room for expansion in order to determine what other record types are required and how many of each type are necessary. A significant consideration, which is almost completely dependent on application requirements, is the type and total number of pool records that are required.

Those portions of the optional software support required by the system must be chosen before system generation. Some examples are unit record support and many of the application support packages. In addition, the users are faced with deciding on options in the given software packages. More significant are the requirements for whether or not the generated system will operate under the control of VM. VM support presents a number of options that are explained separately.

---

## Tightly Coupled Processor Support

TPF tightly coupled (TC) processor support is part of the base product. It provides a single TPF system image when running on processors with more than 1 CPU. By installing a processor with more than 1 CPU and by modifying application programs as described in *TPF Application Programming*, users can achieve greater performance than that available with single CPU processors. TPF tightly coupled support can also be used in combination with the loosely coupled facility of the High Performance Option licensed feature (described below) to achieve even greater levels of performance.

The number of CPUs (also known as instruction streams or I-streams) that the TPF system uses is essentially transparent to the system operator. Existing operator commands operate on the single TPF image.

No new system generation options are required for a TC system. All TPF systems generated are capable of running on processors with 1 or more CPUs. If desired, an operator message can restrict the system to using only a single CPU, but there is no path length benefit in doing this. Generally, the TPF system should be set to allow use of all available CPUs.

TC support provides a load balancing mechanism called the I-stream or tightly coupled scheduler. The I-stream scheduler, when invoked, chooses the CPU on which a message is processed. This scheduler has 2 tuning parameters that you can use to change the way the scheduler behaves. These values may be left at the default setting or may be modified online by operator command. There is no method for setting these values at system generation time. (See “Tightly Coupled Scheduler” on page 48 for more information.)

---

## TPF Communication Support

The TPF system supports the following communication protocols:

- Airlines line control (ALC) support

One of the following is required:

- 3725 communication controller with the Network Extension Facility (NEF) feature installed.
- 3745 communication controller with the Airlines Line Control Interface (ALCI) feature installed.

See *TPF ACF/SNA Data Communications Reference* for more information about ALC support.

- Binary synchronous communication (BSC)

A 3745 communication controller with the 3745 partitioned emulation program (PEP) is required. See *TPF Non-SNA Data Communications Reference* for more information about BSC.

- Systems Network Architecture (SNA)

LU types 0, 1, 2, 3, and 6.2 are supported. See *TPF ACF/SNA Data Communications Reference* and *TPF ACF/SNA Network Generation* for more information about SNA communications.

- Synchronous link control (SLC)

A 3705 communications controller with the emulation program (EP) is required. See *TPF Non-SNA Data Communications Reference* for more information about SLC.



- TCP/IP support

The TPF system supports socket applications through a Transmission Control Protocol/Internet Protocol (TCP/IP) offload device or an IP router. See *TPF Transmission Control Protocol/Internet Protocol* for more information about TCP/IP support.

- X.25 protocol support

One of the following is required:

- A 3725 communication controller with the X.25 NCP Packet Switching Interface (NPSI) feature installed.
- A 3745 communication controller with the X.25 NCP Packet Switching Interface (NPSI) feature or Fast Transaction Processing Interface (FTPI) feature installed.

See *TPF ACF/SNA Data Communications Reference* for more information about X.25 support.

See the *TPF Migration Guide: Program Update Tapes* for a complete list of supported hardware and software.

---

## High Performance Option

HPO contains 2 components: the multiple database function (MDBF) and the loosely coupled (LC) facility. These may be used separately or in combination.

When generating either of these components, you must choose the options needed for the system ahead of time. In either case, the generation procedure involves the use of new macros plus new keywords on base generation macros. MDBF systems require a generation of each subsystem in addition to generating the base system. For LC systems, only one base generation is performed, regardless of the number of processors in the complex.

## Multiple Database Function

The multiple database function (MDBF) provides two basic options, but from a generation point of view there are three user choices:

- In a base system without the HPO feature or in a system with the feature but without MDBF, the system should be viewed as having only 1 subsystem, the basic subsystem (BSS), that contains only 1 subsystem user.
- With the HPO feature you can:
  - Generate a system with only 1 subsystem, the BSS, which may have 1 or more subsystem users; or
  - Generate a system with multiple subsystems, the first of which is the BSS. Any of the subsystems can have multiple subsystem users. A subsystem consists of a unique program base and a unique database operating under a common control program. The basic subsystem also includes necessary system service to support the MDBF environment.

The number of subsystems and subsystem users must be defined before generation. It is recommended that additional subsystems and subsystem users are generated for future expansion. This avoids the need to regenerate the system at a later date. The basic subsystem must be generated first.

A major consideration to keep in mind is that any support required by a particular subsystem must be provided for by the BSS. In addition, any change to a hardware

configuration, for example additional DASD devices, will result in a regeneration of **all** subsystems in addition to the BSS because these devices are defined in the generation of each subsystem. On the other hand, many software support options may be unique to a subsystem and should **only** be defined for the subsystem that requires them.

TPF provides a number of system generation options for MDBF, along with 2 different versions of the cross subsystem access services CSECT, CCCSAS (one for use with and one for use without MDBF). In a non-MDBF system, the MDBF support code is bypassed, reducing work load and allowing ECB-controlled programs to be identical at the source level.

## Loosely Coupled Complexes

For loosely coupled complexes, it is important to understand how hardware and some software is to be treated. All hardware and some software can be grouped into one of three categories:

- Shared
- Processor unique
- Switchable.

All DASDs are shared with access to data records serialized by the use of an external locking facility (XLF) such as the limited lock facility (LLF) or the concurrency filter lock facility (CFLF) on DASD control units (CUs), or a coupling facility (CF). Any processor in the loosely coupled complex can access and update file resident records. Contention is resolved through a combination of system software and the locking hardware. *Processor unique* hardware or software is under the exclusive control of a single processor in the loosely coupled complex. For example, unit record devices are handled as processor unique. Software utilities can be made processor unique through the processor resource ownership table (PROT). Switchable resources are addressable by all of the processors but, in fact, are only used by one processor at a time, often with software-enforced controls. Tape devices are an example of a switchable resource. A loosely coupled user should pay close attention during generation to the category to which a resource belongs.

For generation purposes each of the processors in a loosely coupled complex is generated from a single image (for example, a single SIP input deck). There are very few cases (keywords) where the use of this facility must be uniquely treated but the number of physical processors and the hardware configuration must be known before system generation.

The major differences in generating a non-loosely coupled complex are in defining the hardware and in assigning hardware addresses. Shared and switchable resources are usually assigned the same physical addresses in each loosely coupled processor, and this is done as if there is only one processor. Processor unique resources are assigned different addresses, but again, as if there is only one physical processor. In some cases (for example, unit record devices) the different physical devices can in fact have the same addresses and those addresses are only specified once. In other cases, the physical addresses must be different. Each device type has its own rules that must be followed.

Unique to a loosely coupled generation is the need to define the processor resource ownership table (see data macro PR1OT). This is done using the SIP macro, UTPROT. Another SIP macro, GLSYNC, is used to define the records and fields that are located in the global area, which must be synchronized between all

I-streams in the complex. Use of the globals in this environment is more fully described in *TPF System Installation Support Reference*.

Another major generation consideration of a loosely coupled complex is the requirement for an EP processor *if* there is a need to support 3705 EP or 3270 locally attached devices. It should also be noted that a number of software functions that are only associated with 3705 EP (for example, message switching) are performed only by the EP processor. The first processor in the loosely coupled complex is automatically designated the *EP* processor. See the SYSID parameter of the CONFIG macro.

Users are cautioned to consider all aspects of a loosely coupled complex. For example, an NEF-supported ALC terminal can be considered to be shared by all loosely coupled processors because there is a path to it from all processors, but the user's NCP generation may direct that terminal to send all input messages to only one of the processors.

Users must generate the system to accommodate expansion and collapse of the loosely coupled complex. In addition, consideration must be given to the necessity to switch over any one of the processors in the loosely coupled complex to another physical processor. It is recommended that the system be initially generated with the maximum number of processors that will be required throughout system life rather than forcing a complete regeneration to add additional processors to the complex in the future. Switchover may be accomplished by treating the fallback processor as an additional processor in the complex (for example, defining 3 processors versus just 2 that are really needed, but never actually using 3); or by a hardware configuration where a backup processor can simply replace any existing processor in the complex. The key is to lay out the hardware in a manner that best suits user requirements before attempting to generate a system to support the required hardware configuration.

---

## Multi-Processor Interconnect Facility

The Multi-Processor Interconnect Facility (MPIF) provides an interface that allows TPF systems to perform cooperative processing with other TPF systems. Using MPIF, one TPF system can request a system service that is received, processed, and returned by another TPF system. The requesting system does not need to be aware that the receiving system can be in a different TPF complex.

The following diagram shows a sample MPIF complex consisting of 3 TPF complexes.

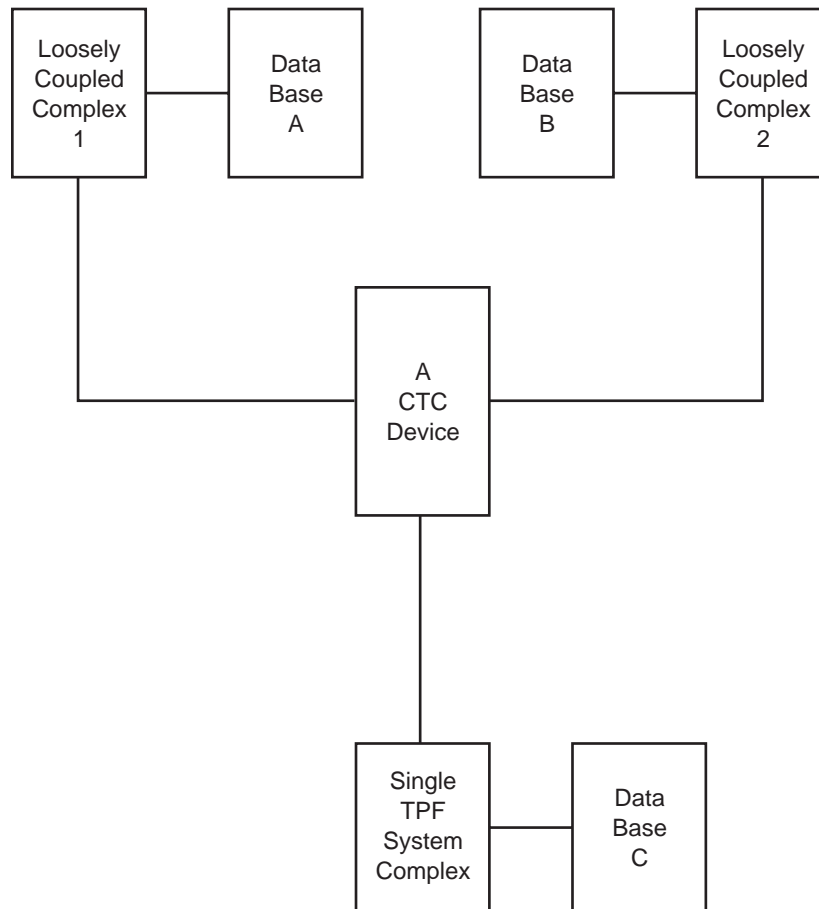


Figure 3. MPIF Complex

Once installed, operator commands are available to define parameters, paths, and devices to be used by MPIF.

Generation considerations include the following:

- Specify YES on the MPIF parameter of the CONFIG macro.
- The RAMFIL macro must be coded for MPIF record types #HDREC, #PDREC, and #PDREU.

---

## Interrelationship of Multiprocessing Components

The relationships of loosely coupled multiprocessing (LC), tightly coupled multiprocessing (TC), interprocessor communications (IPC), and the multiple database function (MDBF) are somewhat complex. A comprehension of these relationships is particularly important in order to install a TPF system. Figure 4 is an abstraction of the relationships. Frequently the term *abstraction* connotes unrealistic, difficult, or obscure. This is not the intent of Figure 4. This abstraction is used to hide many of the details of reality which, if exposed, obscure the principle ideas to comprehend concepts. A familiar expression used to convey this thought is: "You can't see the forest because of the trees." To use a metaphor — Figure 4 is the TPF forest. If you have been patient enough to reach this point, then the exposures should be helpful in order to unravel the relationships, which are frequently hidden in the trees.

Figure 4 includes 4 CPCs, each with 4 I-streams. Each CPC interconnects through channel-to-channel (CTC) communications, several communication controllers, and a sizeable amount of DASD. The communication lines, which are attached to the communication controllers, are equally distributed to the 4 CPCs. There are 2 subsystems, each with 2 subsystem users. There are 4 applications, 1 for each subsystem user. Consider what happens when 16 messages arrive (at about the same microsecond) at each CPC. Through luck and the statistical smoothing algorithm mentioned in “Scheduler Algorithm” on page 50, each message gets dispatched to each of the 16 I-streams a few nanoseconds apart. Through bad luck each message is quadrupled; for example, the messages that cause Entries to be created for the same applications on each of the CPCs result in 16 FIND and HOLDs to the same record. All records are VFA candidates but none are currently in the *cache*, which is full. Furthermore, each of the applications makes extensive use of the global area for *intra* program communication.

Some easy exposures:

1. The messages could all have been for a single application with 16 messages still dispatched to 16 I-streams, necessary for the 16 FIND and HOLDs to be to the same record.
2. Are the FIND and HOLDs recommended? Will the system prevent this?
3. How much interprocessor communications (IPC) traffic will the applications generate among the CPCs?

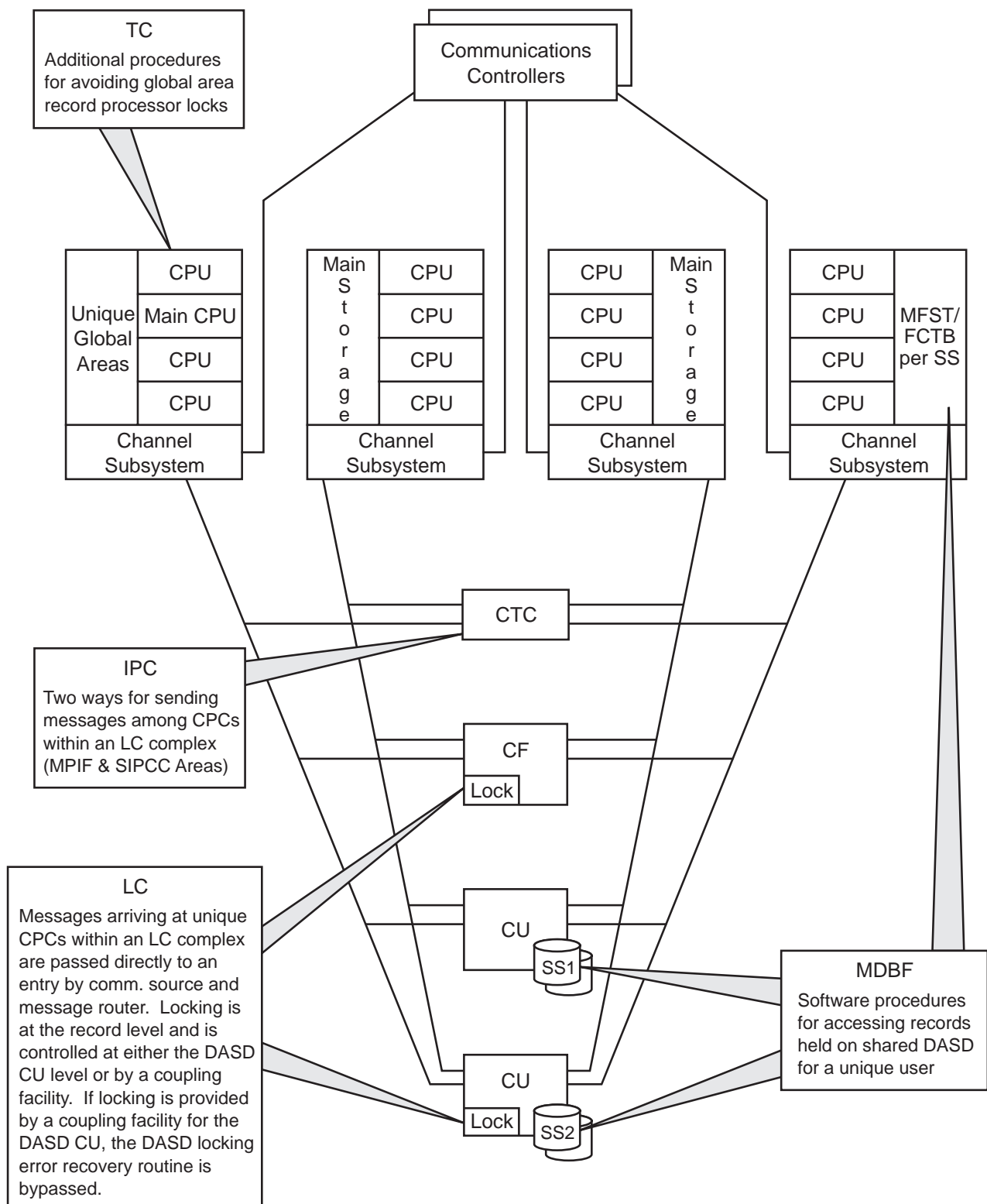


Figure 4. Interrelationship of Multiprocessing Components

---

# System Generation Process

In order to generate the TPF system or any of the communication device operating systems, an MVS operating system is required. In addition, all of the offline support programs are executed under MVS control.

---

## System Initialization Overview

TPF system initialization includes the assembly of the system programs and laying out the database, application programs, system records, and application records. In essence, the system programs are used to manage the resources used by application programs and application records. Procedures in addition to the MVS assembly program are also required. An outline of the initialization process is as follows:

1. Unpack the tapes received from the IBM program information distribution center. This means placing the contents of the tapes on MVS libraries. See the unpacking guide for more information; details are not included here.

**Note:** To perform a system initialization program (SIP) process on the TPFDF product, unpack the tapes received from the IBM program information center. This means placing the contents of the tapes in MVS libraries. See the unpacking guide for more information; details are not included here.

2. Compile and link the FACE table generator. (See "Compiling the FACE Table Generator" on page 188 for more information about compiling the FACE table generator.)
3. Create a SIP stage I input deck.

**Note:** See *TPFDF Installation and Customization* for more information about defining the TPFDF product.

4. Code a RIAT table.

**Note:** See *TPFDF Installation and Customization* for more information about coding a RIAT table when performing a SIP process on the TPFDF product.

5. Code a system allocator input deck that contains the names of any user application code.
6. Run the FACE table generator to create a FACE table and other database-related information.
7. Assemble the SIP stage I deck to create a SIP stage II deck.
8. Run SIP stage II. This creates the keypoints and macros, assembles and compiles the necessary segments, builds the CP, runs the system allocator, and so on.

The TPF system provides the user with the following facilities in order to generate the required communication network records. The first of these (SIP) is always required and the others are optional depending on the communication line disciplines being supported.

- The system initialization package (SIP) contains communication macros and parameters in addition to the host-related macros.
- The communications record generation program (SCKGEN).
- The TPF offline SNA table generation program (OSTG).

- The SNA keypoint (CTK2).

Because all communications are handled by the basic subsystem in an MDBF environment, the communication database is generated only for the BSS and is not relevant to other subsystems.

## System Initialization Package

The system initialization package (SIP) provides a set of macros that are coded by the user to select function and specify parameters relating to the following communication functions:

- Message router
- 3270 Local Support (LC)
- Polling
- Error recovery
- Line configuration
- System consoles
- Non-SNA communication definitions.

Many of the program records SIP produces contain information about the previously mentioned network functions. Some program records are in the form of system keypoints, whereas other program records reside in the program area of the database.

## Non-SNA Communication Keypoint Records

The non-SNA communication keypoint records consist of the communication control unit keypoint status records (PKST) and the system communication keypoint records (SCK). SCK and PKST keypoint records are created by the SCK and PKST generation program as input to the system test compiler (STC). STC creates pilot tapes that are loaded to the online modules by the data loader. These records reside in the fixed file area.

## SNA Communication Tables

The SNA communication tables are defined using the SNAKEY macro. See *TPF ACF/SNA Network Generation* for more information.

## C Language Support

ISO-C support is installed during several phases of the SIP process. TPF C language support, in general, depends on a C compiler and its associated libraries. The C compiler needs to be installed before the TPF system. ISO-C support requires allocation for records that hold or manage C programs. Allocation of records for ISO-C is reflected in RAMFIL statements used to lay out the database and define the FACE table. The amount of storage needed for C programs in main storage is reflected in the CORREQ macro, which handles main storage structure requirements. ISO-C library load modules are created as a part of stage II of the installation process. SIP stage II creates 2 library load modules: the TPF API library CTAL and the standard C library CISO. Procedures for creating load modules are installation-specific and must be created before application programmers can reliably use ISO-C support.



---

## TPF System Loading

Following the assembly of all the system and application programs, and the generation of all the system and application data records, the next step is to load the online files.

### Program/Keypoint Loading

The programs and keypoint records are loaded to the online system packs in a 2-step process that requires a loader general file and online system packs. SIP produces JCL to initialize and format a loader general file. The required number of online files must be initialized and formatted before starting to load the online system. The system loading steps are as follows:

1. The control program, online keypoints, file and main storage resident program segments, and the startup program records are loaded to the loader general file by the offline system loader under control of MVS as shown in Figure 5 on page 18.
2. All of the program segments are loaded from the loader general file to the online packs by IPLing the loader general file, which is under the control of the TPF system.

**Note:** The term *startup programs* refers to those real-time program segments which are really data records and are normally only loaded once, at initial restart. These records are typically modified by the system during execution and should not be reloaded unless the user requires that they be placed in their initial status. An example of these types of records are the application program name records of the message router.

There is a separate loader general file for each subsystem including the BSS in a MDBF environment.

The loading of programs and the correspondence of program names to system storage is done via an allocator and an offline loader.

- The allocator (known as the TPF system allocator) identifies where the program resides on system storage.
- The system allocator (SALO) builds an MVS data set, called the SAL table, containing the names of every program used by the TPF linkage editor (LEDT). The system allocator also builds an online allocation table (IPAT) that is used by the online system to determine program characteristics. There is a separate IPAT for each subsystem in an MDBF environment.
- The system loader is a two-part process; the general file loader and online module load. The offline segment builds the loader general file. (A general file represents a sequential set of data that is accessed through TPF I/O services.)
- To build the loader general file, the offline loader calls on the TPF linkage editor. This linkage editor resolves the external symbols (VCONs to the MVS assembly program), which were used to name TPF application programs. This is done in a manner that is foreign to the normal MVS linkage editor. This is the point all references to application programs are resolved for every application program before the application programs are placed at system resident locations. These references are compatible with the TPF data organization and represent the names of the system storage locations of the programs.
- The system loader online segment uses the loader general file as input to place the online programs at their designated online file locations. The online loader segment runs under TPF control.

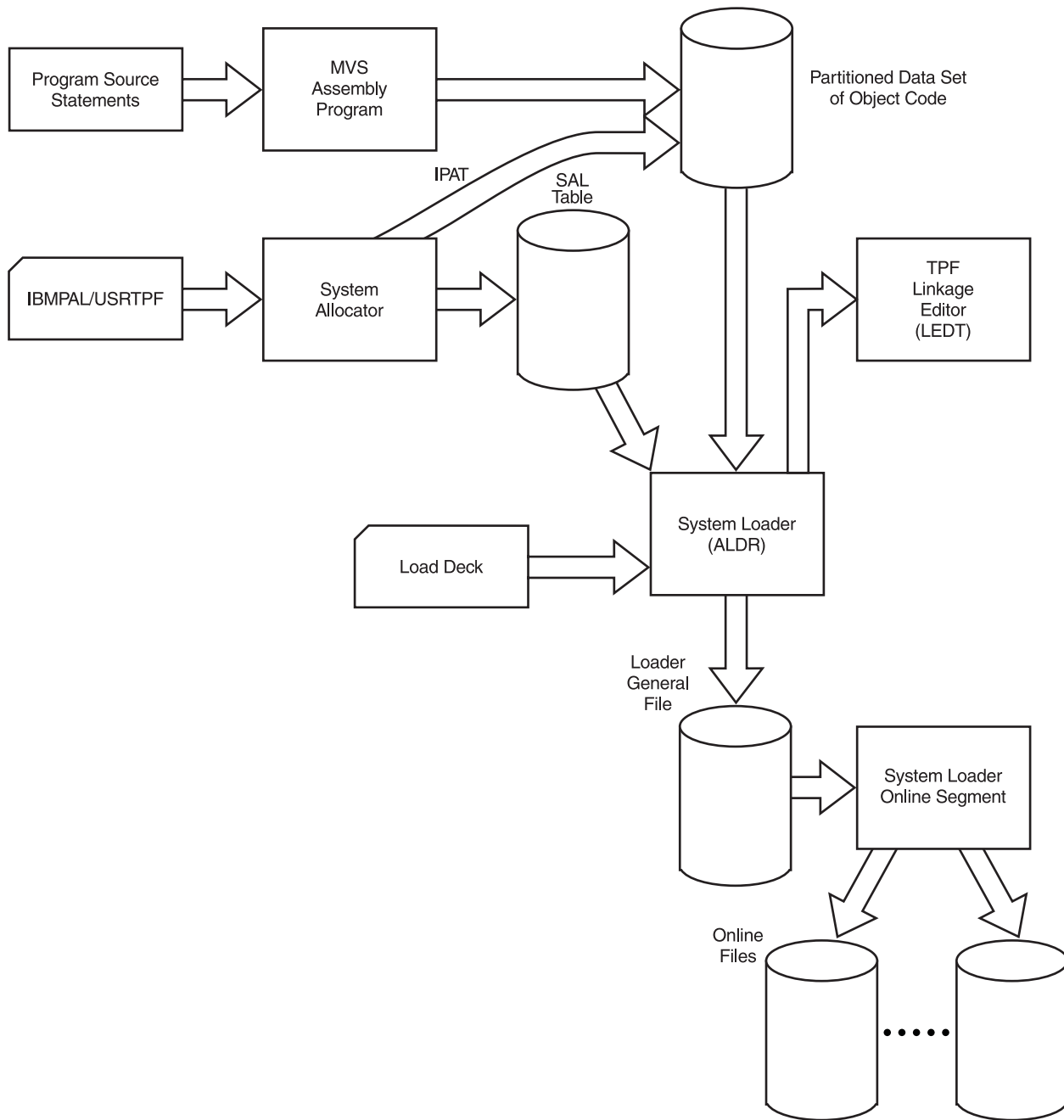


Figure 5. TPF Program/Keypoint Loading

## Loader General File

The loader general file is initialized and formatted by the SIP process. A separate job using the system loader offline segment is then executed under MVS control to create a general file that can be IPLed. The loader general file is created through a rudimentary language that represents the input to the offline loader. The statements, or control cards, of this language are called the load deck.

The relationships between the system loader offline segment, the system loader online segment, and the restart areas on both the loader general file and the online modules are shown in Figure 7 on page 22 and are listed as follows:

1. The offline segment, using the load deck as input, creates a restart area on the loader general file. This restart area is built from data found in MVS partitioned data sets. The restart procedure may be invoked from the loader general file by mounting the disk pack and pressing the IPL button. The online segment, although ECB-controlled, is not allocated through the TPF system allocator.
2. The offline segment, using the load deck and the object library as input, creates data records including the data that is to be placed into the restart area of the online modules.

An existing user must, of course, reconcile the keypoints on the loader general file with those on the existing system. This is done either by not loading keypoints on the loader general file, by modifying the keypoints on the loader general file, or a combination of the two.

3. The loader general file is mounted and the IPL program invoked. The core image records of the TPF core resident control program and the system loader online segment are placed into main storage by the IPL program. The online segment is ultimately given control and executes as an ECB-controlled program. This is an initial restart.
4. The online segment loads the online modules with the input records found on the loader general file. This loading includes the keypoint and core image portions of the online prime restart area. (The backup copies of the keypoints are created during normal online execution.)
5. The online segment also places the file resident programs on the online modules, spreading them across all modules in the record type.
6. The online segment exits to the restart scheduler. When the procedure is completed, and the key system data is loaded through pilot tapes, the system can be IPLed from the online modules. The IPL from the prime restart area is shown in Figure 7 on page 22. The online IPL is necessary because the content of the keypoint and core image portions of the online module restart area is not the same as the corresponding portions of the loader general file. The IPL program must be invoked to place the different information into main storage. As a matter of fact, different values in CTKX and keypoints V and A make the IPL program react differently during an online restart than during an initial restart. Additionally, the online IPL is necessary to activate the interprocessor communications facility (IPC) of the loosely coupled facility.

In summary, an initial restart may be characterized as a system restart, done from the loader general file, which invokes the system loader online segment. An online restart is done from the IPL prime module and does not invoke the system loader online segment.

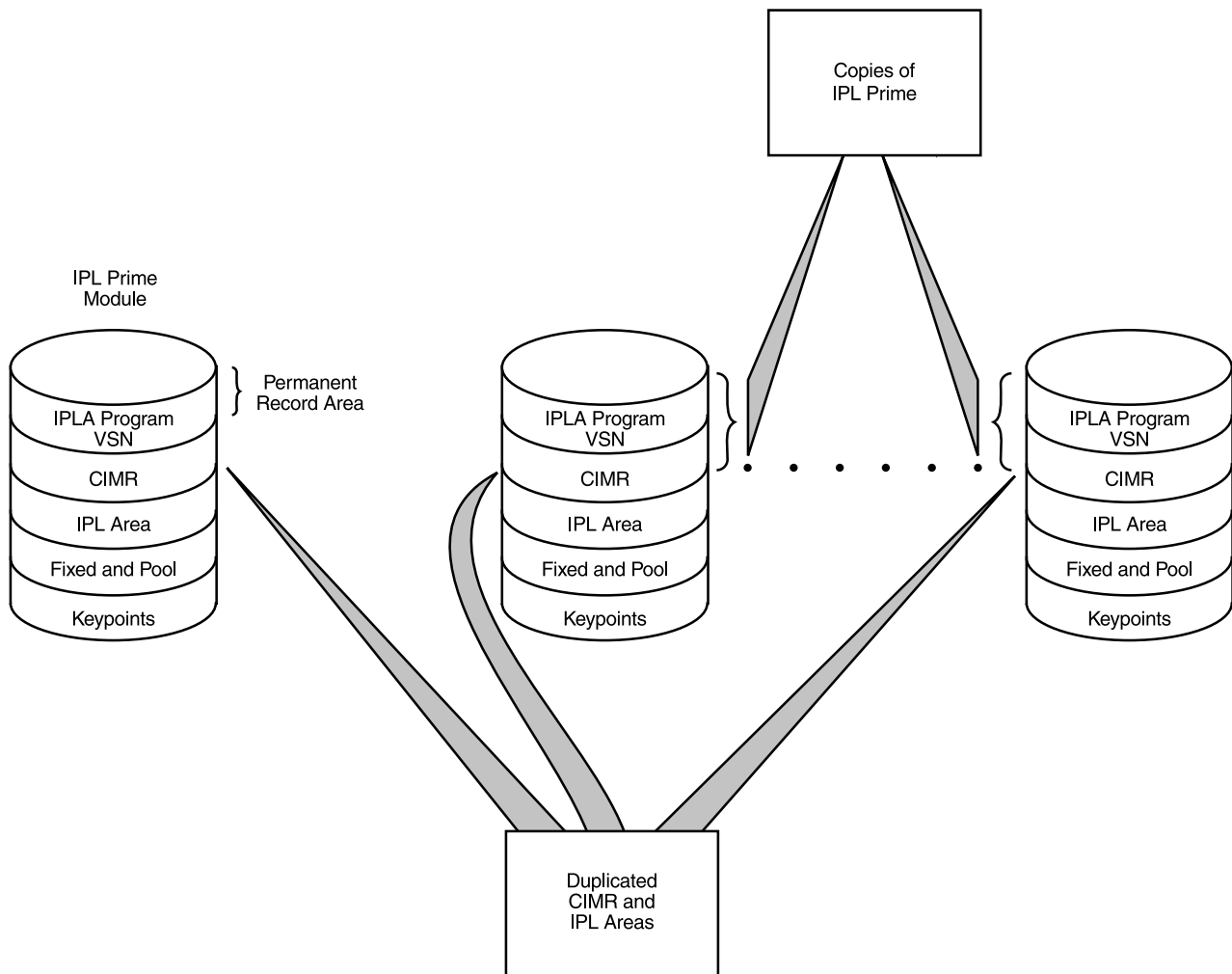


Figure 6. Allocation of TPF Online Modules

## System Restart

The TPF control program is restarted by using the IPL option on the operator control panel of the hardware console (known as a hard IPL) or by using the ZRIPL command (known as a soft IPL). Whenever TPF is restarted, an initial copy of the control program is loaded from the online system packs to main storage. The keypoint records are loaded from the IPLed device and are used to start various fields and tables in main storage. All working storage is initialized with all core blocks appearing on the uncommitted storage lists. When initialization is complete, a message will appear on the operator console.

When the TPF system (through catastrophic error recovery) or the operator starts a soft IPL with the ZRIPL command, the IPL program will decide whether or not to reuse the main storage copy of some system tables and records. This decision is based on data in the IPL program's fast recovery table (FRT). (Refer to the FRORT data macro.) This table contains information on system records, which may be saved across a software IPL. It is used to reduce the amount of data that must be loaded during system recovery to NORM state. This table contains validity information as well as start and end addresses of the FRT records. Also, on a soft IPL, the VFA area in main storage will be reused if the data is still valid.

When a hard IPL is performed, none of the records in the fast recovery table are reused. If there are no VFA delay file pending buffers, the main storage copy of VFA will **not** be reused. If there are delay file pending buffers, the main storage copy of VFA will be reused. This is done to ensure VFA data integrity by allowing the delay file data to be filed after the system is restarted.

**Note:** If a hard IPL is performed with the CLEAR option, all of storage is cleared and there is no way the system can reuse VFA or the FRT records.

System restart consists of 3 components, the goal of each is as follows:

- IPL program (IPL1, IPL2, IPLA, and IPLB) initial program load.
  - Load the control program including the initializer program (CCCTIN), the keypoints, and the core resident tables.
- Initializer program (CCCTIN)
  - Allocate and initialize storage tables and lists.
- Restart schedule, system restart, and state change
  - Place the system in 1052 state.

The IPL program loads the keypoints from the fixed file keypoint area into main storage. Automatic volume recognition, called *disk roll-call*, is invoked to assign hardware addresses to the file devices that must be premounted, and the module file status table (MFST) is initialized. The core image records are used to load main storage, including the initializer. The IPL program passes control to the initializer.

The initializer program, which was link-edited with the core resident control program by the MVS linkage editor, refers to information in the core resident control program. The initializer program (CCCTIN) performs the following functions:

- Assigns values to pertinent low-core locations obtained from the keypoint records.
- Sets up interrupt linkages (PSWs and CP tables).
- Calculates and assigns values to pointers in main storage on configuration and work lists.
- Assigns values to tables used by various portions of the control program.
- Obtains an entry control block (ECB) and transfers control to the restart schedule. (During an initial restart from the loader general file, the system loader online segment is entered before invoking the restart scheduler.)

The restart schedule is found in programs CTKS and CTKO. This is a sequence of ENTERs to ECB controlled system programs which set up system tables used for resource management and system execution. The restart schedule also ensures that devices (for example, real-time tapes) that are required for online execution are available. Many of these programs are related to the system services associated with components to be described in the following chapters. Upon completion, a message is sent to the system console.

For more information, see *TPF Main Supervisor Reference*.

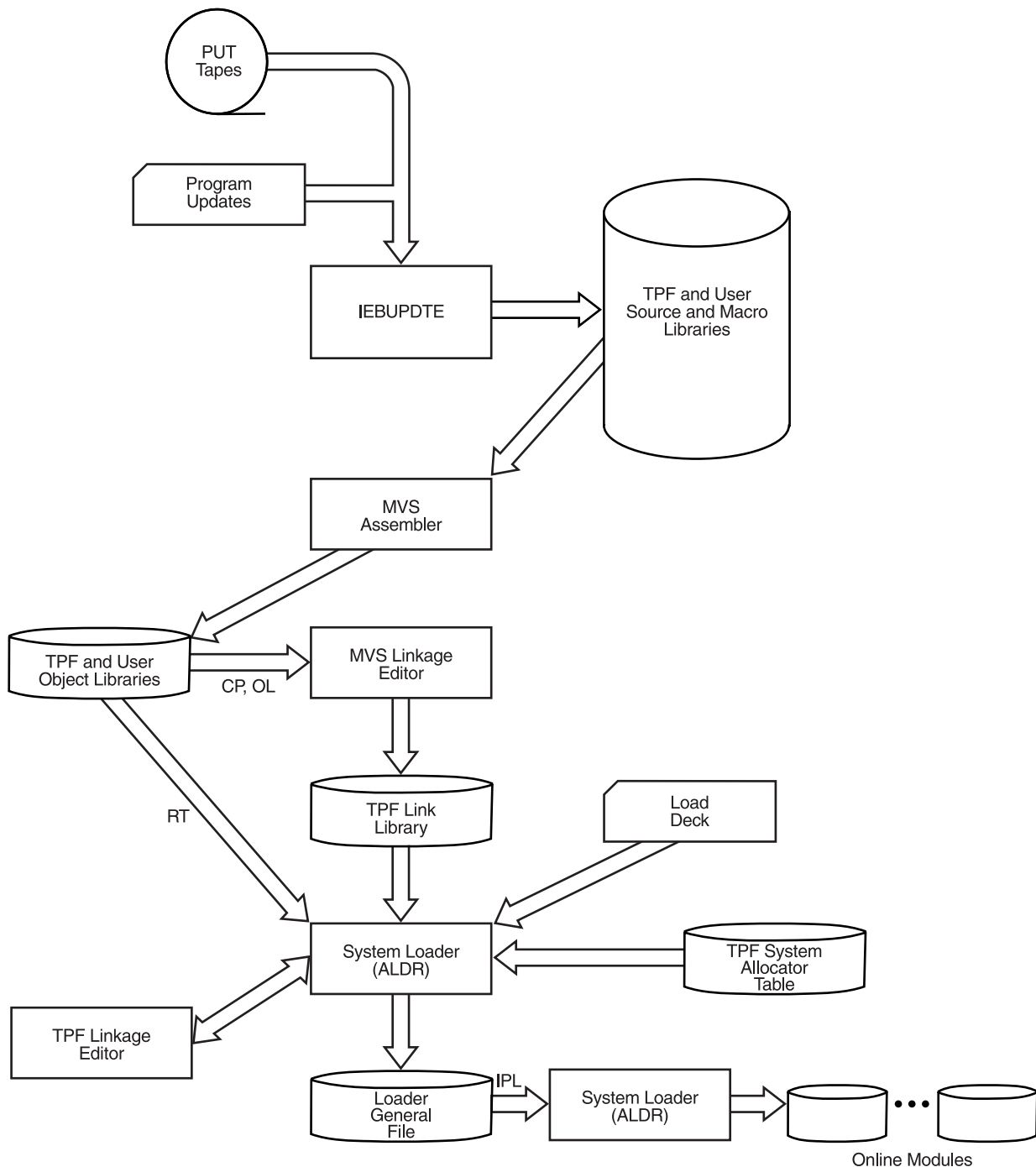


Figure 7. Loading the TPF Restart Area

## Database Loading

The concept of a TPF system state is associated with system restart and the switchover procedure. The precise meaning of the various TPF system states can only be described in the context of other system concepts, not all of which have been described. However, the general concept can be described. A system state is related to the resources managed and function performed by the online TPF system. Some of this resource management and system function is:

- File pool management
- Keypointing
- Communication facilities support
- Program sharing and fetching (ENTER/BACK mechanism)
- Management of system clocks
- Directly attached terminal (computer console) support.

The various TPF system states are related to the function and resource management the system is willing (or able) to support. The two states most easily identified are:

- Computer Console (1052) state
- Normal (NORM) state

The computer console (1052) state means that the system will accept commands from a directly attached computer console. Many system services are not available in the 1052 state, including clock management, file pool management, and communication facilities support. NORM state means all system services are available. The names of the remaining states that fall between 1052 and normal states are listed for completeness. Their precise meanings are not necessary for the current description.

- Utility State

System limitations are imposed to execute certain batch-oriented TPF system utilities (for example, restore the online files). These utilities are usually run during low activity periods. Communications are not active in utility state; input is limited to the prime CRAS.

- CRAS State

CRAS state permits commands only, which may be entered from specially designated terminals that are attached to the communication lines.

- Message Switching State

In this state the system accepts the messages, which are routed to other systems.

See *TPF Operations* and *TPF Main Supervisor Reference* for an explanation of the various states and how to change states. What is important to understand now is that system restart has placed the system in 1052 state, that the system is essentially idle (no communications activity), and that the primary operator console is active. In addition, 3270 local terminals, which are designated as alternate computer room agent sets (CRAS), may also be active if logged to the system message processor. Before the system can permit communications (application) activity by cycling to a state higher than 1052 state, the following database records must be loaded on the online files:

- Pilot tapes
- Pool directories.

Users of the multiple database function (MDBF) of the HPO feature should interpret *system* as *subsystem* in this entire section including “Device-Independent File Addressing” on page 24 and “File Address Formats” on page 25. The term *system* should be replaced by *subsystem* because each subsystem’s database of both fixed and pool records is unique and separate from each other subsystem, including the basic subsystem (BSS). For example, in the preceding paragraph when the system is referred to as being in 1052 state to load the pilot tapes and pool

directories, MDBF users should interpret this to mean that each subsystem's pilot tapes and pool directories should be loaded when that subsystem is in 1052 state.

In addition, any programmable communication devices must be loaded before cycling to a state that permits communications activity.

Pilot tapes can be loaded any time the system is in 1052 state. The data loader (ACPD) loads the pilot system data records to the online disk modules. A header contains the record type and record ordinal that is used by the file address compute program (FACE) to determine the actual file address.

**Note:** If you create the pilot tape with an ID of N, you can load the pilot tape when the system is in any state by using the ZSLDR command. See *TPF Operations* for more information about the ZSLDR command. See *TPF Program Development Support Reference* for more information about creating the pilot tape.

Pilot tapes are required for application data records, the processor resource ownership facility (for details see *TPF Main Supervisor Reference*), non-SNA communication keypoint records, and optionally, SNA communication tables.

A user converting an existing system to a new one may not be loading pilot tapes. In this case, any embedded fixed DASD addresses must be in FARF format. Pool file storage usually comprises a large portion of the application database, plus pool files are required for several system functions (unsolicited message processor (UMP), long message transmission (LMT), and others). Pool file directories are used by the online system to control dispensing of the pool records. See *TPF Database Reference* for more information about pool file directories.

## Device-Independent File Addressing

Device independent file addressing in TPF applies only to DASD devices.

The objectives of device-independent file addressing are:

1. To implement a file address format that is independent of physical device characteristics for DASD devices and future secondary storage devices.
2. To preserve all current application interfaces to the database.
3. To extend current data addressing facilities, principally the number of fixed file record types per system and the number of records in each type.

The objectives are met by implementing the following:

1. A file address reference format, called FARF, is the device-independent addressing scheme used in TPF systems. FARF3 means file address reference format 3, FARF4 is format 4, FARF5 is format 5, and FARF6 is format 6. The file address reference formats are those bit configurations placed into the file address reference words (FARWs). The main distinction between reference formats 3, 4, 5 and 6 is the amount of disk storage that can be addressed by each. With FARF5, a fullword is used for addressing, while in FARF3 and FARF4, less than a fullword is used, implying addressing restrictions. With FARF6, two fullwords are used for addressing.
2. All existing application interfaces are preserved. Fixed records are accessible through an entry to the FACS program with record type and ordinal number; FACS returns a 4-byte file address in FARF format; and a FIND/FILE macro is then issued with the FARF file address on the appropriate ECB data level. For data event control blocks (DECBs), file records are accessible by using the FAC8C or FACZC macro to obtain an 8-byte file address; and a FIND/FILE



macro is then issued with an 8-byte file address on the appropriate DECB. For pool records, get file storage and release file storage (GFS/RFS) macros remain the same. All commands applicable to the database have also been preserved.

3. For FARF3 only, data addressing is a maximum 67 108 863 records per pool type and 268 435 456 fixed records per system. In the fixed records, there may be up to 12 252 record types, each of which may contain up to 268 435 456 records.

FARF4 addressing has a maximum of 1G addresses (1G equals 1 073 741 824) and FARF5 addressing has a maximum of 4G addresses, (4G equals 4 294 967 296) with no fixed percentage devoted to fixed records or pools. FARF6 addressing has a maximum of 64 petabytes, or PB (64 PB equals 72 057 594 037 927 936).

## File Address Formats

TPF file address formats are described in *TPF Database Reference*.

## Implementation

All support required for online FARF addressing is automatically created by offline generation of the FACE table and pools tables. The user's main concern is with the SIP RAMFIL statement because this forms the basis for the TPF database.

In FARF3 formatted addresses, the BAND parameter of the SIP RAMFIL macro allows the assignment of one or more subrecord types (band numbers) to a fixed record type. Users should code a unique band number, from 0–4095, for each 64K, or part thereof, of fixed records. Note that these band numbers will be physically associated with each band of 64K records in the sequence of the users SIP input.

It is important that although the FARF3 format is device independent, it is band number dependent. It is for this reason that band numbers, by design, are assigned by the user rather than automatically generated by SIP. The intent is for users to maintain the same band number assignments from one TPF system generation to another. This ensures that any embedded FARF3 addresses will always refer to the same logical record even though the record may have been physically relocated or even assigned a different record type.

FARF4, FARF5, and FARF6 formatted addresses use a scheme involving a universal format type (UFT) and a format type indicator (FTI). The UFT portion of an address selects a section of DASD address space and the FTI portion selects a record type within that section. FARF4 and FARF5 are UFT/FTI dependent in the same way that FARF3 is band number independent. UFT/FTI pairs defined for FARF6 are independent from UFT/FTI pairs defined for FARF4 and FARF5.

**Database Reorganization:** DBR allows the user to reorganize all or selected fixed and pool record types. See the *TPF Database Reference* for more information. Also, within a record type, ordinal number ranges may be specified. All these DBR options are input to the system via commands.

The output phase of DBR captures the appropriate records to tape. It may be run in 1052 or NORM state.

Before running the DBR input phase, the user should perform whatever procedures are required. (For example, reformatting the online files, generating a new FACE table, reassembling the allocator, etc.).

The DBR input phase reloads the database in two steps. Fixed records appear on the capture tape (DBF) and are loaded during the general file IPL. Pool records

appear on the capture tape (DBP) and are loaded once the prime module IPL is complete. Both steps are triggered by commands.

## Communication Device Loading

The IBM 3705 device, which is no longer in production, is the only communication device that can be loaded by the TPF system.

Use the 3705 communication controller system support package to load the emulation program (EP) load modules to the IBM 3705 device. See *TPF Data Communications Services Reference* for more information.

---

## Post-Cutover Activities

Once the new system is operational, it must be maintained, whether it is a test system or a production system. APAR fixes (normally distributed via PTF tapes) need to be applied on a continual basis and application programs and database records may require updating on the same basis.

System modifications are normally done in 1052 state. It is highly recommended that all changes be first validated in a test system prior to putting them in an operational system. The following facilities may be used to aid in this effort:

- Alter file and alter core operator commands.
- New pilot system records may be loaded at any time in 1052 state.
- The general file loader enables the loading of new versions of the resident control program, start-up program segments, keypoints, and/or E-type (real-time) programs. It utilizes both offline and online segments plus a loader general file. Figure 8 on page 27 describes the entire program maintenance process when it is utilized.
- The E-type loader enables loading of new programs or new versions of existing programs in any system state.
- The auxiliary loader enables the loading of the same types of programs as does the system loader described above but operates in 1052 state or above and does not require the use of a loader general file and its associated IPL. The new programs are passed to the online system via tape input.
- Multiple TPF image support allows user's to perform system loads in NORM state without destroying the existing program base, and to immediately switch a processor to a previous program base without requiring a loader general file IPL. For more information about these facilities, see *TPF System Installation Support Reference*.





---

## **Part 2. Defining the TPF System**



---

## Processor and Main Storage Resources

This chapter describes how TPF utilizes main storage, including its organization and allocation. This chapter also addresses processor support, magnetic tape support, and unit record support. All system generation in this area is provided by SIP.

---

### Processor Definition

For the purpose of this discussion, the term *processor* means any system comprised of 1 or more central processing units (CPUs). A CPU is a device capable of executing a program contained in main storage. With the introduction of tightly coupled processor support, two new terms have been introduced. The first is CPC (central processing complex). The second is instruction-stream engine (or simply, I-stream). CPC is synonymous with processor, while an I-stream is a CPU.

TPF may utilize any of the enterprise systems architecture (ESA) processors when a non-loosely coupled system is desired. A loosely coupled TPF system may use any extended architecture processors that have been equipped with the required RPQs. (See *TPF Migration Guide: Program Update Tapes* for a summary of supported processors and RPQs.)

As many as 32 processors can may be connected together using the loosely coupled (LC) facility of the HPO feature. Each processor runs a separate image of TPF, but all processors share the same database. See “Loosely Coupled Complexes” on page 10 for an additional explanation of LC.

Some multiprocessor models are partitionable. Such a processor may be used either as a single integrated processor, or as independent processors.

If you decide to use both processor partitions in the same loosely coupled complex, and clock synchronization is being handled either through the TPF TOD RPQ or the Sysplex Timer (STR) with a TSC then each logical processor requires a unique Time-of-Day (TOD) Synchronization Selection Address (SSA). (See “CONFIG” on page 219 for information on how to code SSAs.) Systems using STR without the TOD RPQ or TSC do not use SSAs. A unique SSA is required because each partition is considered to be a separate processor. There is a maximum of 8 SSAs, one for each of the first 8 processors in the LC complex. Additional processors beyond the first 8 should not be assigned an SSA value.

The number and types of channels and control units, and the number and types of I/O devices, are determined during the design of the system. SIP provides the CONFIG and IODEV macros to describe the processor and channel characteristics, respectively.

The following parameters of the SIP CONFIG (processor configuration) macro relate to processor support. (For more information about the CONFIG macro, see “CONFIG” on page 219.)

- APRNT—address of the system printer for use during the IPL sequence.
- VM—whether or not the TPF system, normally a test system, is to be run under the control of the VM program product (for example, VM/SP).
- SYSID—the symbolic TPF processor ID(s).
- PROC1–PROC32: These parameters are coded only when an LC system is required. Each parameter specifies the serial number and the model number for a processor in an LC complex. If the TOD clock synchronization RPQ is installed,

then each PROC parameter contains a subparameter specifying the TOD Synchronization Selection Addresses for a processor.

**Notes:**

1. Many of the required initialization parameters in this section are associated with a SIP parameter. SIP parameters are indicated where appropriate, with the SIP parameter given in parentheses preceding or following the macro name. For example:

SIP Input = (SYSID) CONFIG, or CONFIG (SYSID)

Means:

SYSID is a parameter used within the SIP macro CONFIG.

2. For users of the LC facility, multiple symbolic processor IDs are specified via the SYSID parameter, one for each of the processors in the LC complex.

In addition to the hardware TOD clock, TPF provides for software clocks which are available to application programs. For users of MDBF in the HPO feature each subsystem is allowed to have its own version of these software clocks. Each subsystem's clock values are calculated using the hardware TOD clock. Non-MDBF users are treated similarly to a BSS in an MDBF system.

These software clocks are located in low memory. Keypoint record A (CTKA) contains the subsystem's local standard time difference from Greenwich Mean Time (GMT). Keypoint record B (CTKB) contains the subsystem's local standard time (LST), the subsystem's GMT, and the subsystem's perpetual minutes clock. There is a copy of these keypoints for each subsystem defined at system generation.

SIP macro parameters that relate to the software system clocks (CLOCKS) are as follows:

<b>GDIF</b>	System clock LST minus GMT.
<b>LST</b>	Precycle above 1052 state LST of system clock in EBCDIC.
<b>DATE</b>	Precycle above 1052 state date of the system clock.
<b>DELTIM</b>	Initial system time difference from the system clock.

**Notes:**

1. *Precycle above 1052 state* means the value to be used by the subsystem prior to cycling the subsystem above 1052 state at which time the system operator can change the value using commands.
2. The MDBF user will have to code multiple CLOCKS macros, one for each subsystem. The non-MDBF user only codes one CLOCKS macro. This technique of providing different user values by subsystem in a MDBF generation is the standard one used by SIP as is the concept of only coding one version of a given macro in a non-MDBF environment.

Support for the generation of multiple subsystems, each with its own database (DASD devices), is provided by the multiple database function (MDBF) of the HPO feature. There can be a maximum of 64 subsystems, including the basic subsystem (BSS). Generation of an MDBF system is indicated by coding the SSDEF SIP macro. (For more information about the SSDEF macro, see "SSDEF" on page 312.)



---

## TPF Main Storage Layout

The organization of main storage for program and data manipulation is the key to the design of a high-performance system such as TPF. Storage protection is used for fixed storage to prevent an application from illegally modifying programs and critical data records (located in the global area). Part of main storage contains programs and data that are common to all entries and that always remain in main storage. This portion is called fixed storage or permanent core in TPF. Another portion, called working storage, is allocated to entries as required. Programs and data are transferred from file storage to working storage as required. Fixed storage consists of the following main storage areas:

- Resident control program area
- Core resident application program area
- Application global area
- Control program records and table area.

Each individual area of both fixed and working storage will be expanded upon in later sections.

A general layout of main storage is provided in Figure 9 on page 35. Starting from low core (shown at the bottom of the figure), each main storage area is addressed, up to the 16 megabyte line, by subsequent subsections of this chapter.

Algorithms are provided to calculate some system definition values.

The APSIZ parameter of the CORREQ macro relates to the overall layout of main storage. APSIZ specifies the size of the core resident programming areas above and below the 16M line.

The MDBF user will have to code multiple CORREQ macros, one for each subsystem. The non-MDBF user only codes one CORREQ macro.

Rather than calculate the sizes of the various areas by hand, users should make a rough pass at selecting the various main storage related SIP parameters.

Automatic resizing of the fixed storage areas takes place during an IPL.

Storage protection keys are included in the main storage layout in Figure 9 on page 35. This provides a summary of storage sizes for the various mandatory and optional program and data areas in main storage. Values for the parameters may be calculated or estimated as explained in the accompanying sections. Actual sizes may be obtained from SIP Stage II listings and, for SNA support, the updated and assembled SNA keypoint (see data macro CK2SN).

Use of the word *core* as in *core storage*, *core image*, and so on, relates to the System/360 generation of computers. The correct terminology for System/370 and later systems is *main storage*. The words *core* and *main storage* are used interchangeably in this publication.

In the event of errors, or on request, TPF provides a system error dump of memory for the purpose of problem determination. The amount of storage dumped is controlled by using the commands ZDSER and ZASER. The amount of storage dumped can also be determined by the use of overrides created during the IPL (see the IDOTB macro in *TPF System Macros* and *TPF Migration Guide: Program Update Tapes* for more information) and by using the ZIDOT command.

---

## TPF Control Program Area

The TPF core resident control program is assembled and link-edited by SIP. The link-edited version is known as CPS0vv, where vv is its version number, and it is assigned to main memory to control the online execution of the system. It also resides on the online files for the purpose of system restart in a location known as the core image area. It consists of a set of required and optional control sections (CSECTs). Which CSECTs are included is based upon user selection of SIP macros and macro parameters for the particular system configuration being generated. Several CSECTs are released as object code and SIP only reassembles these if requested by the user. Typically, each CSECT, in some cases more than one, supports a major functional area (for example, tape support, unit record, SNA, and so on).

The control program serves all subsystems in an MDBF environment. It is part of the BSS. The size of the control program is dependent upon the system configuration selected. Some of the CSECTs are optional and are only required if related hardware is included in the system configuration.

The size of many of the individual CSECTs of the control program, both optional and mandatory CSECTs, is also influenced by the fact that Basic Assembler Language (BAL) conditional assembly language (the AIF statement) is also used to include/exclude lines of code and tables. SIP first produces PDS members SYGLB and SYSET which contain global variable statements/values and are then copied (via the assembler COPY statement) by the segments of the various CSECTs when the control program is assembled by SIP. These values are obtained/calculated from user-coded SIP input. The purpose is to resolve user/installation variables at assembly time.

The control program nucleus employs the following main storage boundaries:

- Less than 4K (page zero)
  - This area requires no base register. It contains PSWs and constants.
  - Also contained within this area, at fixed locations and for fixed sizes, are required fixed-storage machine-check logout areas, and pointers to the various system stack areas.
- 4K to 12K
  - This area contains the primary nucleus code and is addressed by base registers R11 and R12 (that is, X'00001000' and X'00002000').
  - Typical of the code in this area are linkages to other control program CSECTs, often in the form of post-interrupt routines, the first level I/O interrupt routine, and other critical routines such as the fast-link macro decoder.
- Above 12K
  - This area contains code addressed by variable base registers.

The size of the control program is approximately 970K.

At the end of each CSECT is a macro that rounds each CSECT. This allows the next CSECT to start on an easily identified boundary and provides a patch area within each CSECT.

There is also a particular CSECT, CPLKMP, which contains a table which has the name including version number of each CSECT and its starting address and length.

This CSECT is provided to give easy access to this information. It appears in a dump of the system for use when debugging.

Storage Area	Protection Key	Storage Area	Protection Key
System Heap Storage	C	ISO-C Stack Storage For Threads (Area 2)	1
(never mapped in SVM)		System Heap Storage	C
Page 0s	F	(never mapped in EVM)	
CIO Code/Blocks	F	Page 0s	F
FACE, RIAT, etc.	F	CIO Code/Blocks	F
31-Bit Core Resident Program Area	F	FACE, RIAT, etc.	F
PAT, XPAT, etc.	F	31-Bit Core Resident Program Area	F
Extended Globals	F	PAT, XPAT, etc.	F
Global Areas for other I-Streams	C, 1, C	Extended Globals	F
SVM Page/Seg. Tables	F	(never mapped in EVM)	C, 1, C
Assorted Tables	F	SVM Page/Seg. Tables	F
VFA Storage (Buffers and Control Tables)	F	Assorted Tables	F
ECB Page/Seg. Tables	F	VFA Storage (Buffers and Control Tables)	F
CLH Tables	F	ECB Page/Seg. Tables	F
IOBs	F	CLH Tables	F
SWBs	F	IOBs	F
ECBs		SWBs	F
		ISO-C Stack Storage For Threads (Area 1)	1
		ISO-C stack	1
		ECB Heap	1
16MB		16MB	
4K Frames		ECB Private Area (1M)	
4K Common Frames	varies	4K Common Frames	varies
24-Bit Core Resident Program Area	E	24-Bit Core Resident Program Area	E
Control Program Records and Tables	F	Control Program Records and Tables	F
I-S Unique Global Areas GL1, GL2, GL3	C, 1, C	I-S Unique Global Areas GL1, GL2, GL3	C, 1, C
I-S Shared Global Areas GL1, GL2, GL3	C, 1, C	I-S Shared Global Areas GL1, GL2, GL3	C, 1, C
Control Program Area	F	Control Program Area	F
Low			
IPL and System Virtual Memory		ECB Virtual Memory	

Figure 9. Virtual Storage Layout

**Note:** TPF's virtual address spaces makes it impossible to use a full 2 gigabytes of real memory.

---

## Application Program Areas

Application programs, including system supplied programs that are ECB controlled, may reside in one of the following program areas:

- Application program area (31-bit above 16MB)
- Application program area (24-bit below 16MB)
- DASD.

The TPF release includes a number of ECB controlled system programs. These programs plus the core-resident control program CSECTs comprise the TPF online system programs. SIP employs a program dependency table to eliminate those programs and CSECTs which are not required for the functions selected via SIP Stage I macros.

Disk space for programs is defined by a SIP RAMFIL statement allocating #PROGx records using the RECNO parameter. All program records are 4K. Space for programs in main memory is specified by the APSIZ24 and APSIZ31 parameters of the CORREQ macro.

During Stage I execution, SIP processes the SPPGML macro to determine which programs and versions must be assembled, link-edited, and loaded during Stage II.

The program tables (SPPGML) are divided into 12 separate groups:

- Offline (OL) programs
- Real-time (RT) programs
- Control program (CP) CSECTs
- Keypoints (KP)
- Real-time special group (RTS)
- User real-time programs (UR)
- Object code only programs (OCO)
- C language real-time programs (CRT)
- C language user real-time programs (CUR)
- ISO-C assembler functions (ISA)
- ISO-C source segments (ISC)
- ISO-C load modules (ICL).

In addition to the program name, each entry contains a version number, if applicable. The size of these tables is preset at a maximum limit based on the needs of the current release and an estimated number of user real-time programs. You may need to modify or add to these tables before Stage I.

IBMPAL is a file containing a list of released programs and other structures that must be allocated as part of the TPF system. Each entry is in a format that can be run directly into the allocator SALO. IBMPAL statements are also used to allocate transfer vectors, spare slots, and GFS IDs. For ISO-C, only ICL entries (libraries or dynamic load modules (DLMs)) require corresponding PAL entries. ISA and ISC entries do not require PAL entries.

Most IBMPAL statements are program allocations. Programs can be allocated as either core resident, indicated by CR, or file resident, indicated by FR. The older TPF notions of *corefast* and *coreslow* are no longer meaningful. Certain

ECB-controlled programs (for example, those that are really data records and those with non-reentrant code) are allocated as file resident.

All assembler programs reside on DASD as 4KB #PROG records. ISO-C programs reside in 1 #PROGn record and 2 or more #XPRGn records, which are allocated as 4KB. Therefore, the programs may be larger than 4KB when loaded into main storage.

SALO generates the program allocation table (PAT), which is read during IPL. The PAT (IDSPAT) serves as the online system allocator. It also contains the file and main storage addresses of all E-type programs. PAT information is used by CTIN in carving out the core resident areas.

Assembler and TARGET(TPF) C programs are loaded to #PROG records. Each ISO-C program requires a #PROGn record and at least 2 #XPRGn records. For ISO-C, the #PROGn record contains the ordinals of the #XPRGn records and the #XPRGn records contain the compiled program text and ADCON relocation information. Both #PROGn and #XPRGn records are program version record (PVR) unique. #XPRGn records need to use the RESTORE=NO parameter on their RIAT definitions. This ensures that they are not restored.

The class of a program can be any one of the following: shared, private, IS-unique, common, or unprotected. When restart is processing the PVRs and it encounters a C load module, it changes the core copy of the PAT to always show the class attribute as shared, although the file copy of the PAT still contains the original allocation. This is not a problem; DLMs and libraries that are allocated with different characteristics do not need to be changed.

The amount of memory allocated to program areas is affected by input to the SIP CORREQ macro. The APSIZ24 and APSIZ31 parameters in the CORREQ macro estimate the amount of main storage to be used for core resident programs, often application programs. A program base requires #PROGn, #XPRGn, and #PVRn records.

RAMFIL statements control the amount of space available on each DASD device. The RECID parameter is set to the various record types (#PROG1–8) for both TPF-supplied and user-supplied programs. The RECNO parameter specifies the total number of slots to reserve on disk for image x.

Programs allocated to the application program area are ECB-controlled programs assigned to main storage in the online environment. These programs are classified as core resident programs.

They are loaded into one of the core resident program areas (CRPAs) in main storage either above or below the 16M line depending on whether their MODE is 31-bit or 24-bit. Both user application and system file resident ECB controlled programs may reside in the program areas. C load modules, dynamic load modules (DLMs), and libraries all run in 31-bit mode.

Programs in the core resident program area are accessed via ENTER-type macros.

A core resident program adheres to all the reentrant (recursive) conventions implied by the enter-back mechanism. Programs with anticipated high activity are classified core resident and assigned to main storage. Several entry control blocks (ECBs) may be associated with the same reentrant core resident programs.

The size of the core resident program areas has a direct bearing on system performance. This is quite understandable because the larger the main storage area the greater the number of programs which can reside in main storage. Consequently, the number of accesses to files for programs will diminish in an inverse proportion to the number of core resident programs allocated.

For systems with a large amount of available main storage and a large application, it is desirable to have at least the number of core resident applications times 4 KB of the main storage allocated to the core resident program areas. Data collection/reduction should be used to modify program allocations for optimum performance.

The APSIZ parameters of the CORREQ macro are used to estimate the size of the core resident program areas in 1 KB blocks. A minimum of 1000 or more blocks must be specified for the required CORREQ core resident system programs, and the two areas (24-bit and 31-bit) together must be at least 1000 blocks.

A list of the released programs that should reside in the core resident program area can be found at the beginning of the IBMPAL deck.

The standard C library and TPF API libraries used for ISO-C reside in the core resident program area (CRPA).

Reserve a portion of CRPA space for user C programs. Once you enter a ZOLDR ACCEPT or ZOLDR DEACTIVATE command and a C load module is no longer needed (because no ECBs will execute it), the storage used by the C load module is released.

You need a #PROGn record for each C language program along with at least 2 #XPRGn records. A formula can be used to estimate the approximate number of #XPRGn records needed. Each directory record can hold approximately 500 entries and each program record is 4KB.

HPO feature users of MDBF should be aware of the fact that program allocation is done separately for each subsystem, with the BSS done first. In effect, this results in a core resident program area for each subsystem. The core resident program areas are laid out sequentially, by subsystem. For example, the size of each subsystem's area is estimated via the APSIZ parameter of the CORREQ macro for each subsystem. The size required by the system is the sum of the sizes of the subsystems, both above and below the line.

One method for estimating a value for APSIZ is:

"Total Area" divided by 1024 = APSIZ

This should be done for each core resident area, one above the 16M line and one below the line.

Immediately following the core resident programs above the 16M line reside the following programs.

*Table 1. Application Program Areas above the 16M Line*

Program Name	Notes
General File Loader Online Segment (ACPL)	After running SIP, verify the size of ACPL by looking at SKGTSZ from your SIP Stage II deck. (Size is approximately 20 KB.)

Table 1. Application Program Areas above the 16M Line (continued)

Program Name	Notes
FACE Table (FCTB)	After running SIP, verify the size of the FCTB by looking at SKGTSZ from your SIP Stage II deck.
In-Core Dump Formatter (ICDF)	After running SIP, verify the size of the ICDF by looking at SKGTSZ from your SIP Stage II deck. (Size is approximately 85 KB.)
Record ID Attribute Table (RIAT)	The size of the RIAT depends on the number of RIATA macros coded. Each RIAT entry requires approximately 20 bytes of storage.
Program Allocator Table (PAT)	The size of the PAT depends on the number of program, transfer vectors, and spares allocated. Each PAT entry requires approximately 88 bytes of storage.

The SIGT is built by SIP as a result of input from the user-coded GLSYNC macro and there is a separate table built for each MDBF subsystem if both facilities are required plus a common table for all subsystems. For non-MDBF users there is only one SIGT for the entire system.

The general file loader online segment (ACPL) resides in main storage only during the online load operation (for example, when the loader general file is IPLed). This program is not in main storage during online operation.

The FACE table (FCTB) contains a list of base file addresses which the FACE program uses to compute a file address. There is a separate FACE table for each subsystem in a MDBF system. See Offline FACE Table Generator for more information.

The record ID attribute table (RIAT) contains information on every file type within the data base. This information includes pool record attributes (such as size, duration, and duplication), VFA candidacy for fixed- and pool-file records, logging, exception recording, user exit, and record caching attributes. There is a separate RIAT table for each subsystem in an MDBF system.

The in-core dump formatter (ICDF) is optionally used to print system error dumps directly to the online printer but only when the system is in test mode.

The size of the SIGT varies based upon the SIP input. Each subsystem's table is comprised of 2 subsections, section 0 and 1.

Section 0 contains an 8-byte header and an 8-byte item for each subsystem user (SSU) in an SS. Therefore, the size per subsystem is 8 times the number of SSUs plus 8.

Section 1 has an 80-byte item per synchronizable global for each SSU. For each SS a maximum of 256 global fields and 256 global records may be defined as synchronizable (see "Application Global Area" on page 40). For example, if there are 2 subsystems, the first with 2 SSUs with 8 synchronizable fields and 2 synchronizable records, the second SS with 6 SSUs with 30 synchronizable fields and 6 synchronizable records, you must allocate space for 2 SIGT section 1 tables as follows:

- SS1 allocation:  $8 + (8 \times 2) + 2(80 \times 10) = 1624$  bytes



- SS2 allocation:  $8 + (8 \times 6) + 6(80 \times 36) = 17\,336$  bytes

The total size of the application program area can be estimated by adding the number of core resident programs times 4096 to the sizes of ICDF, if selected, ACPL, FCTB, and SIGT if required.

---

## Application Global Area

### SIP for Globals

#### GLOBAL Macro

GLOBAL is a SIP macro that initializes fields in keypoint A to define the size of the I-stream shared or unique areas. The GLOBAL parameters must be large enough to accommodate all the globals that users intend to load from the pilot tape. Once all the STC input is coded for the pilot tape, users must calculate how many 4K blocks will be needed for each of the I-stream shared or unique areas of both primary and extended globals. The result of the calculations must then be used as the parameters for the GLOBAL macro.

When manually preparing records for STC input you can designate global records as primary area resident or extended area resident by setting the global attribute byte of the GOA to the desired option. Likewise, records that are designated I-stream shared must have the correct indicator set in the Global Attribute byte.

Where a Global resides is essentially *transparent* to users. The only requirement an application program must observe is that it must be in a 31-bit mode to access extended globals.

#### Extended Globals Considerations

The SIP GLOBAL macro, mentioned previously, is used to select extended globals. Parameters SSUXUNQ1, SSUXUNQ2, and SSUXUNQ3, are used in the same way as SSUIUNQ1, -2, and -3 to define the number of 4096-byte blocks desired for the three respective extended global areas for the main I-stream. Three other fields, SSUXSHR1, SSUXSHR2, and SSUXSHR3, provide the corresponding information as SSUISHR1, -2, and -3 for application I-streams.

If **all** such parameters are omitted or set to zero, the CCCTIN system initialization program generates a system with no extended globals.

If **any** extended global is defined, an extended globals area will be carved by CCCTIN in upper storage just below the CIO area. Its size will depend on the GLOBAL parameters just described.

GLOBAL also initializes the fields in keypoint A that indicate which load mode each subsystem user will load. This load mode is also stored in the Subsystem User Table entry for each subsystem user. When GOGO is retrieving GOA records, it will only load the entries that reside in a GOA whose GO1NUM field is equal to the Load Mode specified in the Subsystem User Table entry for the loading subsystem user.

#### GLSYNC Macro

To allow a global to be synchronizable, users must code a GLSYNC macro in the Stage I deck for each global field or record to be so designated.



**Note:** All I-streams in an LC or TC complex must load the same file address for a particular synchronizable global record. If a synchronizable global record is I-stream unique, each I-stream must also load the same file address for that record.

For more information about globals, see *TPF System Installation Support Reference*.

---

## Control Program Records and Table Area

The control program records and table area in main storage is allocated and partially initialized by the control program initializer (CCCTIN) as part of the restart procedure. CCCTIN uses information contained in system keypoint records. These keypoints are generated by SIP based upon user input data provided via the SIP macros.

Numerous tables and data records are contained in this area. SIP Stage II calculates the size of each table and outputs the total size of the entire control program and table area as part of the SIP-created macro SKGTSZ.

Various records and tables are discussed in many of the later sections of this document. For example, in the sections concerning communications support ("Data Communications Support" on page 123) there is a discussion of the particular requirements of those functions upon this area. The user is advised to read all of these sections prior to making the estimate. The alternative is to use the SIP report which results from the execution of Stage I. The user must also be aware of the effects of activating the SNA support which is accomplished by updating the SNAKEY macro in the SNA communications keypoint (CTK2).

In addition to the other records and tables, which are discussed in later sections, there are several that can now be considered.

The record hold table is a set of 2 tables (primary and overflow) containing 16-byte entries used for controlling the record hold facility. This facility allows an entry block exclusive use of a record. Both tables reside in the control program records and table area and are allocated by the initializer program. SIP provides the PHTBL and the OHTBL parameters of the RAM macro to specify the number of primary and overflow tables entries respectively.

---

## Working Storage

The size of working storage is a function of the amount of main storage required by an average message, the life of the message in the system (ECB life), and the number of messages per second to be processed. The elements allocated are entry control blocks (one per entry), programs required from file storage, data blocks either created in main storage or transferred from file storage, and message blocks.

Working storage is divided into five areas: 4K common blocks, 4K frames, ECBs, SWBs, and IOBs.

Input/output control blocks (IOBs) are blocks created by the control program to schedule, control, and contain the information required to perform I/O operations. IOBs are located in main storage in the control program records and table area. The SIP macro CORREQ contains an IOBLK parameter which is used to specify the number of I/O blocks required.

System work blocks (SWBs) are blocks created by the control programs. MPIF uses SWBs for output message queuing and control program user exit parameter lists. MPIF-IPC uses SWBs to track usage of core blocks for long SIPCC messages. SNA uses SWBs for scheduling postinterrupt routines.

SWBs are located in main storage in the control program records and table area. The SIP macro CORREQ contains an SWB parameter that you use to specify the number of SWBs required. (For more information on SWBs, see “Storage Required For Control Blocks” on page 46 and “CORREQ” on page 229.)

For the MDBF user of the HPO feature the allocation of working storage is done by subsystem (SS), that is, there is a CORREQ macro for each SS including the basic subsystem (BSS). The considerations for each SS are the same except for one additional concern for the BSS, which also affects non-MDBF users. When allocating the 4K blocks in the BSS, the internal trace table uses either one or two 4K blocks, and there must be at least seventeen 4K blocks allocated for keypoints. In a non-BSS subsystem, there are a minimum of eight 4K blocks required for each SS's unique keypoints.

The following design criteria may be used as a guide to determine the number of each size block required for a particular application and system message rate. If the total amount of main storage is insufficient to allocate the necessary amount of working storage, the user should redesign his main storage layout by altering the number of programs in the application program area (see “Application Program Areas” on page 36).

Users should be aware that reducing the number of core resident programs increases the number of core blocks required for program storage and file accesses, which increases the message life and, consequently, increases the system working storage requirements.

Another factor that could affect the amount of working storage needed is the program nesting level. One can assume that the more programs are nested, the more working storage will be needed unless programs being entered are already in main storage. Users must determine the best main storage balance for his system.

The amount of main memory required for each message is application dependent and difficult to calculate. It can be observed on an operational system by using system monitors and data collection runs where this data is used to verify the design, tune the system, and make projections. Therefore, for this document, a default value of 4700 bytes per message will be used for mean working storage size. To provide sufficient working storage, a message arrival rate peaking factor of 2.3 can be used; this represents a 99 percentile standard deviation above the mean value. The estimated working storage requirement per message then becomes  $4700 \times 2.3 = 10\,810$  bytes per message.

The life of the message in the system is really the life of the ECB and consists of the processing time plus database access time. What this means is that 9200 bytes/message are tied up for the duration of time expressed as the ECB life.

The term *messages* as it is used in this section refers to the system message rate in messages per second.

The following formulas may be used to estimate the initial number of working storage blocks required for a particular application:

Let: S15 = Total working storage required (See Figure 9 on page 35)  
 S = Maximum memory required for each message  
 M = Message life  
 R = Messages per second  
 C = CPU process time per message  
 A3 = Access plus queue time per DASD device  
 F3 = Number of accesses to DASD per message  
 I = Number of instructions per message  
 p p<sup>1</sup> = CPU model performance = instructions per second

There is also a 4K storage block requirement for interprocessor communications (IPC) in a loosely coupled environment:

- The minimum IPC requirement for 4K blocks in a complex that will have a maximum of m active processors at any one time is:
  - m–1 blocks for receive side processing, plus
  - m–1 blocks for transmit side processing
  - for a total of 2m–2 blocks.
- The recommended initial allocation includes an additional m–1 blocks for transmit processing for a total of 3m–3 blocks.
- The actual requirement depends on the amount of IPC activity, the frequency of high-priority communications, and the size of the IPC items communicated.

Assumptions:

Channel utilization for DASD device is no more than 50% of the rated capacity.  
 AAA/RCB records must be brought into high-speed memory (VFA).

p p<sup>1</sup> = 2 280 000 instructions/sec  
 I = 14 600 instructions per message (average)  
 S = 10 810 bytes of memory per message (maximum)  
 A3 = .09 sec  
 F3 = 4.5

Then:

C = I/P = (14600/2280000) = .0064 sec.  
 M = C + F3 × A3 = .0064 + (4.5 × .09) = .4114 sec.  
 S15= S × M × R = \_\_\_\_\_ bytes

Working storage may be computed for a design objective of attaining 80 messages per second, for example, as follows:

$W^{80} = S \times M \times R$   
 = (10 810 × .4114 × 80)  
 = 355 779 bytes (maximum load)

This estimate provides a general idea of main storage requirements (S15). It can be checked against the default sizes for main storage structures.

Table 2. Main Storage Structure Sizes

Main Storage Structure	Default Number	Default Storage (bytes)
4KB frames	200	819 200
Common frames	50	204 800
12KB ECBs	100	1 228 800

Table 2. Main Storage Structure Sizes (continued)

Main Storage Structure	Default Number	Default Storage (bytes)
IOBs	100	25 600
SWBs	100	50 400
24-bit core resident programs	75	76 800
31-bit core resident programs	1000	1 024 000
<b>Note:</b> The size of an ECB is fixed at 12KB.		

The requirements for these values must be considered in the light of site dependent factors, such as anticipated message rate. Users of TPF 3.1 may use the percentage numbers from that system as guidance. As general rules-of-thumb some sites consider an adequate number of 4KB frames to be 10 times the number of ECBs; the number of IOBs is slightly smaller and the number of SWBs is larger than in TPF 3.1 because some functions formerly using IOBs are using SWBs.

The size of the ISO-C stack, ECB heap, and the ECB private area of 1 MB can affect the number of frames and ECBs allocated by the TPF 4.1 system. The size of the area occupied by the frames and the ECBs must be greater than or equal to the maximum size of the ISO-C stack plus the maximum size of the ECB heap plus the size of the ECB private area of 1 MB.

The following equation can be used to evaluate if you have enough storage:

$$\#ECBs \times 12K + \#Frames \times 4K \geq (ESPS + EMPS + 1) \times 1M$$

The TPF 4.1 system will increase the number of ECBs and frames until this condition is satisfied or the system runs out of storage.

The numbers of system structures required are entered into SIP using the parameters of the CORREQ macro.

These numbers are starting points only and will require tuning after the system is in use and data collection/reduction has been executed and analyzed. One strategy for defining main storage requirements is to let the parameters default. The values may be adjusted online using the ZCTKA command.

## Storage Requirements for Pool Directory Buffers

This section describes the preliminary storage requirement formulas for pool directory buffers. The calculations are based on set size values. The amount of storage carved by CTIN for directory buffers will increase as the set size increases.

For each long-term pool section defined in the system, the amount of storage carved is:

$$(ICY7CSZ + (2 \times ICY7SSZ)) + (2 \times (CY3SYZ + CY4SYZ) \times \text{set size})$$

For each short-term pool section defined in the system, the amount of storage carved is:

$$(ICY7CSZ + (2 \times ICY7SSZ)) + (2 \times 4K \times \text{INT}(((CY3SYZ + CY4SYZ + ICY8SIZ) \times \text{set size}) / 4K) + 1)) + (2 \times 4K \times (\text{INT}((CY2LDR - CY2FDR) / CY\$CNT) + 1))$$

Both calculations are rounded to the next 4K boundary.

The following table shows the meaning of the labels found in the formulas, and identifies the DSECTs where the values of the fields can be found.

Label	Description	DSECT
ICY7CSZ	The size of the common area of the Pool Directory Set Control Area.	ICY7PR
ICY7SSZ	The size of one buffer control area of the Pool Directory Set Control Area.	ICY7PR
CY3SYZ	The size of one pool directory.	CY3DR
CY4SYZ	The size of the memory-resident pool directory control fields.	CY3DR
ICY8SIZ	The size of one slot in the Short Term Limits Control Subtable.	ICY8CS
CY2LDR	The ordinal number of the last pool directory in a pool section. (This value must be provided by the user.)	CY2KT
CY2FDR	The ordinal number of the first pool directory in a pool section. (This value must be provided by the user.)	CY2KT
CY\$CNT	The number of directory time stamp entries that will fit in one data record of the Short Term Pool Directory Control Record.	CY\$CR

## Storage Requirements for Multi-Processor Interconnect Facility

The following describes the preliminary storage requirement formulas for the MPIF product. The formulas have not been put through any formal validation process; therefore, they are subject to change as experience is gained.

### Storage Required For Tables

TABLE NAMES:

<b>DCTCDB</b>	Connection Definition Block (CDB)
<b>DCTCWA</b>	MPIF CCW Area Table (CWA)
<b>DCTDNT</b>	Directory Update Notification Table (DNT)
<b>DCTGFN</b>	Global Function Name Directory (GFND)
<b>DCTMGT</b>	MPIF Global Table (MGT)
<b>DCTPAN</b>	Path Activation Notification Table (PAN)
<b>DCTPDT</b>	Path Definition Table (PDT)
<b>DCTRFN</b>	Resident Function Name Directory (RFND)
<b>DCTSCT</b>	System-to-System Connection Table (SSCT)
<b>DCTTRC</b>	MPIF Trace Block (MTRC)

For the formulas that follow:

C = compute the storage requirement as per algorithm.

HL = header length.

- IL = item length.
- RS = required storage which can be given or have to be computed using the formula.
- V1 = number of systems connected to this system.
- V2 = number of systems connected to this system plus one (1).
- V3 = number of MPIF users which might be active in this system.
- V4 = number of MPIF users which might be active in the attached processors.
- V5 = number of paths which might be attached to this system.
- V6 = number of CTC communication links that might be attached to this system.
- V7 = number of addresses per CTC communication link.
- V8 =  $V1 \times V3$
- V9 = number of connections which will be made in this system. This has to include 1 CDB per path for the MPIF to MPIF connection plus the number of connections anticipated for each user in this system.
- V10 =  $V6 \times V7$
- X = variables, refer to description for the given label.
- V11 = number of processors in a loosely coupled complex.
- V12 = V11 times the maximum number of paths between processors in a loosely coupled complex.

Formula:  $(X \times IL) + HL = RS$

Item	IL	X	HL	RS
DCTCDB	120	V9	16	C
DCTCWA	160	V10	8	C
	32	V6	0	C
DCTDNT	80	V8	16	C
DCTICD	24	V12	8	C
DCTIGT	56	V11	56	C
DCTGFN	40	V4	16	C
DCTMGT	0	0	0	376
DCTPAN	72	V8	16	C
DCTPDT	216	V5	16	C
DCTRFN	120	V3	16	C
DCTSCT	66	V2	16	C
DCTTRC	0	0	0	4096

## Storage Required For Control Blocks

### DCTMRB (Multi-System Request Block)

The term multi-system request block (MSRB) describes MPIF's use of system work blocks (SWB) to hold input and output data. SWB is a generally used core block type. MSRB refers to MPIF data in an SWB that fits the DCTMRB format. The following formulas calculate the storage requirements for MPIF's use of SWBs.

The number of SWBs required by MPIF is calculated as follows:

$I\_0$  = number of SWBs required for the I/O activity.  
 $N$  = number of SWBs required for the I/O activity and the miscellaneous MPIF requirements.  
 $RS$  = the number of bytes of storage required for SWBs.

(For computation of  $I_0$ , use the larger of the following two values.)

$I\_0 = ((\text{queue depth}) + 5) \times V5$   
or  
 $I\_0 = .10 \times (\text{\# of 381-byte blocks} + \text{\# of 1055-byte blocks} + \text{\# of 4096-byte blocks}) + 20$   
 $(1.5 \times V6) + I\_0 = N$

The amount of storage required by the MPIF feature for SWBs is calculated as follows:

$1024 = RS$

## Storage Required for Staging Buffers

The following factors are important in calculating the storage size required for staging buffers:

1. Each MPIF path is made up of 2 physical links: one is used for reading and the other for writing.
2. The size of the staging buffers can vary by link. The staging area size is forced to a doubleword boundary.
3. A control area size is added to the user-specified size for the staging buffer of a path. This control area size is equal to 1024 bytes, which is the size of an SWB.
4. Two staging buffers are required for read links while 1 staging buffer is required for write links.

## Sample Case for Storage Requirements

THE CONFIGURATION:

Three loosely coupled complexes interconnected together

1 has 6 processors (sample system is in this complex)  
2 has 6 processors  
3 has 2 processors

SWBs: 111 616

Assuming:

$I\_0 = 200$   
 $N = 218$

STAGING BUFFERS: 552 960

Assume user-specified size for all paths is 4096.  
 $1024 + 4096 = 5120$   
 $3 \times V5 \times 5120 = 3 \times 36 \times 5120 = 552\ 960$

TABLES: 46 168

$V1 = 10$  systems  
 $V2 = 11$  systems  
 $V3 = 2$  number of users in this system  
 $V4 = 24$  number of users in the complex  
 $V5 = 36$  number of paths attached to this processor  
2 each to 5 LC systems for IPC  
2 each to 13 systems for any-to-any connectivity  
 $V6 = 4$  3088s  
 $V7 = 32$  devices per 3088

```

V8 = 26
V9 = 59 number of connections
V10 = 128

DCTCDB = 7096
DCTCWA = 20616
DCTDNT = 2090
DCTGFN = 976
DCTMGT = 376
DCTPAN = 1888
DCTPDT = 7792
DCTRFN = 496
DCTSCT = 742
DCTTRC = 4096
*****
* TOTAL CORE REQUIREMENTS FOR SAMPLE CASE 710 744
*****

```

---

## Tightly Coupled Scheduler

### Function

In TPF's tightly coupled implementation, ECBs are assigned to a specific I-stream. They execute only on that I-stream unless software explicitly moves them to another I-stream, even if the I-stream they are using is heavily used and another I-stream has no work at all. For maximum performance, all I-streams must have roughly the same amount of work to do. The tightly coupled scheduler is the tool used to balance the load between the I-streams.

The scheduler is invoked when a SWISC (change I-stream) macro is issued with a target I-stream number of zero. During input message processing, the message router issues a SWISC with a target I-stream number of zero to pass control to the input message processing program. This is how input messages are assigned to an I-stream.

In some cases users may be forced to run certain user application programs only on a single I-stream, generally because the application program has not been written to operate as a reentrant program. Users can bypass the scheduler by using the COBC user exit program to explicitly specify the I-stream number for an input message. COBC is invoked by coding the COMEXIT=YES parameter on the MSGRT macro during system generation or by using the ZSYSG command. See "Communications Source Message Router and System Message Processor" on page 148 for additional information.

### Tuning the Tightly Coupled Scheduler

The scheduler attempts to keep adjusted I-stream utilization in balance across all I-streams. Adjusted I-stream utilization is the percent of the time that an I-stream spends working for all tasks minus the percent of time spent working on tasks that give up control via the DEFRC or the DLAYC macro. Adjusted I-stream utilization is reported directly to the user in the data collection/reduction I-Stream Summary Report (see *TPF System Performance and Measurement Reference* for more details), and can be displayed online via the ZSTAT U operator command. Adjusted I-stream utilization is used instead of actual I-stream utilization to keep looping low priority tasks from taking over an I-stream. (When a task loops using DEFRC or DLAYC, the actual I-stream utilization goes to 100%. If the scheduler used actual utilization then no further work would be scheduled on that I-stream.)



There are two conflicting requirements that must be met to obtain optimal balance: the scheduler must react instantly to any change in overall system workload composition, and the scheduler must ignore changes in load that are the result of random actions rather than an actual workload composition change. For example, if a utility program is started on a particular I-stream then the scheduler must reduce the number of tasks that are assigned to that I-stream to keep the system in balance. On the other hand, short-term changes in I-stream utilization caused by the variance in the amount of I-stream utilization required by input messages must not cause the scheduler to take any action.

The REACT parameter of the ZCNIS command determines how quickly the scheduler responds to changes in load. The higher the REACT parameter is set, the more slowly the system will respond to changes in work load. If the REACT parameter is set too high, then the system will be slow to shift the allocation of tasks in response to a change in workload composition. This condition can be seen by using the ZSTAT U command to display the size of the READY and INPUT list. During normal operation, each I-stream will maintain a certain number of tasks on each of its CPU loop work lists. Following a change in workload composition, such as starting a utility, the number of tasks on the CPU loop lists should change smoothly and reasonably quickly to a new steady-state value. If, following a change in workload composition, the scheduler has caused a workload imbalance by reacting too slowly (REACT set too high), then one or more I-streams will show substantially more tasks on its CPU loop lists than are seen in normal operation. The number of tasks will peak, then slowly start to work down to its normal steady-state value. Once the number of tasks on the CPU loop lists is back to normal, the I-stream workloads are back in balance, and will stay balanced until the next change in workload composition.

If the REACT parameter is set too low, then the system will react to statistical fluctuations in the workload, and will overreact to changes in workload composition. Both of these conditions are characterized by large READY and INPUT list queues that appear on an I-stream, remain for a few seconds, and then show up on a new I-stream. In general, when the workload composition is constant, the ratio between the sizes of the lists on the I-stream should stay roughly the same. If the ratio between the sizes of the lists swings wildly then the I-streams are not correctly balanced.

The user should exercise great care in changing the REACT parameter from the default value. It is safer to set the REACT parameter a little too high than it is to set the parameter too low. Abrupt changes in the REACT parameter may cause unexpected side effects which can lead to poor scheduler, and overall, system performance. If REACT must be changed, the change is best done in many small steps rather than one large step.

The MAXQ parameter provides a necessary safety valve which keeps the system in balance following large changes in workload composition and at extremely high utilization (>95%). MAXQ is a user-specified value that is compared to the number of tasks that are on the CROSS, READY, INPUT, DEFER and VCT lists of an I-stream. Once the system exceeds the MAXQ value on one I-stream, that I-stream will no longer have jobs assigned to it by the scheduler. (The user's ability to assign work directly to an I-stream is not affected.) If 2 or more I-streams hit MAXQ then the scheduler's reaction depends on other scheduling values (see "Scheduler Algorithm" on page 50). In general, the MAXQ parameter should be set so that it is not reached during normal steady-state operation, but it can be reached quickly if the system workload changes.

**Notes:**

1. The ZCNIS REACT and MAXQ parameters should be set based on data observed during normal peak time operation. The scheduler is designed to operate more effectively as the overall system utilization increases. Data obtained during periods of low utilization will not help tune the scheduler for periods of high utilization. During periods of low utilization the CROSS, READY, INPUT, DEFER and VCT are too small to demonstrate the cycling behavior caused by too small a REACT parameter, the sluggish balance adjustment caused by too high a REACT parameter, or to set the correct value of MAXQ.
2. The amount of work that is assigned to an I-stream by the user affects the ability of the scheduler to balance the system. The scheduler does not need to schedule all work to balance I-stream utilization, but it does need enough work to get the system in balance. If the user schedules about the same amount of work to all I-streams, then the scheduler will not need to schedule a substantial amount of work to bring the system into balance. If, however, 95% of the work in a system is assigned directly by the user to a single I-stream, then the scheduler can only use the remaining 5% to attempt to balance the system. It will not do a very good job.
3. The scheduler depends upon accurate utilization data and a granular workload mix. Operation of a TPF system under VM without dedicated I-streams can lead to wildly fluctuating I-stream utilization data caused by VM time slicing of guest virtual machines. During testing, test drivers can generate workloads that change quickly and unpredictably; scheduling one item can cause a substantial amount of work for a long period of time on one I-stream with few or no other pieces of work to schedule on the other I-streams. In general, do not expect to be able to obtain a stable balance in a small scale test environment.

## Scheduler Algorithm

The tightly coupled scheduler has two components. The first component, the monitor, is invoked once a second. It determines I-stream utilization and adjusted I-stream utilization for each of the available I-streams. The monitor then sets routing weights based on the adjusted I-stream utilization and a reaction time constant set by the user. The routing weights are used to schedule messages over the next one-second interval. The second component, the scheduler, uses the routing weights set by the monitor, and a MAXQ value provided by the user, to actually assign an I-stream.

### I-Stream Utilization Computation

Once a second, raw I-stream utilization (RAWU) and adjusted raw I-stream utilization (ARAWU) are computed using the exact elapsed time (ELAPSE), the time spent idle (IDLE), and the time the system spent working on tasks that gave up control via either the DEFRC or DLAYC macro (DTIME). The following equations show how raw I-stream utilization and adjusted raw I-stream utilization are computed:

$$\text{RAWU} = \frac{\text{ELAPSE} - \text{IDLE}}{\text{ELAPSE}} \quad \text{ARAWU} = \frac{\text{ELAPSE} - (\text{IDLE} + \text{DTIME})}{\text{ELAPSE}}$$

I-stream utilization (UTIL) is computed by exponential smoothing using the previously computed utilization (OLDU) and the raw measured I-stream utilization (RAWU):

$$\text{UTIL} = ( (.8) \times \text{OLDU} ) + ( (.2) \times \text{RAWU} )$$

Adjusted I-stream utilization (AUTIL) is computed in the same way using previously computed adjusted utilization (AOLDU) and the raw measured adjusted I-stream utilization (ARAWU):

$$\text{AUTIL} = ( (.8) \times \text{AOLDU} ) + ( (.2) \times \text{ARAWU} )$$

Both I-stream utilization and adjusted I-stream utilization are reported in the I-stream Summary Report produced by the data collection package and can be displayed online via the ZSTAT U operator command.

### Routing Weight Computation

Routing weights are initialized to 100. Once a second the I-stream with the highest adjusted I-stream utilization and the I-stream with the lowest adjusted I-stream utilization are selected. The routing weights for these I-streams are then adjusted to change the relative amount of work that these 2 I-streams receive. Even on a processor with more than 2 I-streams, only the weights for the I-streams having the maximum and minimum adjusted utilization are changed.

The following formulae give the new weights for the I-streams that had the minimum and maximum adjusted I-stream utilization:

$$\text{MIN.WT} = \text{MIN.WT.P} \times \frac{\text{MAX} + \text{MIN} + \text{REACT} + 1}{\text{MAX} + \text{MAX} + \text{REACT} + 1} \quad (\text{ROUNDED DOWN})$$

$$\text{MAX.WT} = \text{MAX.WT.P} \times \frac{\text{MAX} + \text{MIN} + \text{REACT} + 1}{\text{MIN} + \text{MIN} + \text{REACT} + 1} \quad (\text{ROUNDED UP})$$

where:

#### **MIN.WT**

The weight that will be assigned to the I-stream with the minimum adjusted utilization.

#### **MAX.WT**

The weight that will be assigned to the I-stream with the maximum adjusted utilization.

#### **MIN.WT.P**

The weight assigned during the previous period to the I-stream with the minimum adjusted utilization.

#### **MAN.WT.P**

The weight assigned during the previous period to the I-stream with the maximum adjusted utilization.

#### **MAX**

The maximum adjusted I-stream utilization found in last period times 1000.

#### **MIN**

The minimum adjusted I-stream utilization found in last period times 1000.

#### **REACT**

A tuning parameter which changes the rate at which the weights change for a change in utilization.

### Scheduling Tasks

A total field (TOTAL) is maintained for each I-stream. These fields are initialized to zero. When the scheduler is invoked to select an I-stream for a task, each I-stream's total field is checked to select the I-streams with the lowest and second

lowest total. The I-stream with the lowest total is chosen tentatively as the I-stream on which the task will run. The selected I-stream has its total field increased by its weight. (The weight for an I-stream is computed once a second as described above.) The number of tasks on the CROSS, READY, INPUT, DEFER and VCT lists for the selected I-stream is then compared with the user specified MAXQ value. If the number of tasks queued on the CROSS, READY, INPUT, DEFER and VCT lists is less than the MAXQ parameter, the scheduler returns the selected I-stream number to the caller.

If, however, the number of tasks on the lists exceeds MAXQ then the scheduler selects the I-stream with the fewest tasks queued on its lists. The scheduler then returns the I-stream number to the caller.

### **User Specified Scheduling Constants**

Two constants are available to tune the action of the scheduler. The ZCNIS MAXQ message sets the value MAXQ described in the section above. The ZCNIS REACT command sets the REACT parameter given in Routing Weight Computation. Refer to “Tuning the Tightly Coupled Scheduler” on page 48 for additional information.

---

## **Tape Support**

The TPF system requires tape support for online and offline processing. Depending on your installation, your tape configuration and options may vary. The following information will help you:

- Plan your tape configuration
- Configure tape devices
- Customize tape processing options
- Customize stalled tape module queue monitoring
- Remove support for a tape format
- Allocate additional working storage for LIFO devices
- Define default categories for the tape library dataserwer
- Define a minimum tape level for the tape library dataserwer.

## **Planning Your Tape Configuration**

The total number of tape drives required by the online system depends on the configuration of the system and the characteristics of the application programs being run. At a minimum, you need the following real-time tape drives for online use:

- One primary real-time tape drive, designated RTA, for transaction logging and collecting other dynamic system information for offline maintenance and data reduction procedures.
- One logging real-time tape drive, designated RTL, for recording storage dumps resulting from system errors.
- At least one critical record logging tape drive, designated RTx, for logging critical application records.
- At least one tape drive as backup, to prevent a system stoppage caused by the unavailability of any of the tapes just described.

Additionally, you should provide a minimum of 3 general tapes for utility functions. These tapes can be used for either input or output.

IBM recommends that you provide at least 8 tape drives, switchable between 2 control units, to prevent a system stoppage in the event of a tape drive or control

unit failure. There is a degree of flexibility allowed when specifying tape drive requirements, but fully evaluate the proposed system operation before trying to minimize the number of required tape drives. When designing the tape configuration, take into account the need to do a switchover to another physical processor. Design a configuration that allows both the online and the backup processor to have a path to the same control units.

For loosely coupled users of the HPO feature, treat the requirements for tape drives and control units for each processor in the LC complex as if each processor was a uniprocessor. The exception to this is that a control unit may be shared by more than 1 processor in the complex. The tape devices themselves are then assigned to a particular processor via command (ZTVAR). This assignment may be changed dynamically when no processor is physically using a particular device. The LC user, in addition to the uniprocessor requirements for switchover, must also take into account tape configuration when expanding and collapsing the LC complex.

## Configuring Devices

SIP macro IODEV is used to configure channel and control unit information that is used to generate program segment COSY (Tape Control Unit Cross-Reference Table). The tape control unit cross-reference table (TCXR) is required by the restart and reconfiguration functions. The table is read into storage during tape restart.

The following parameters of SIP macro IODEV (device characteristics) relate to tape support:

- IOADR (logical device address)
- DVTYP (tape).

For more information about the IODEV macro see "IODEV" on page 264. For a list of supported devices, models, and features, see the *TPF Migration Guide: Program Update Tapes*.

The device numbers specified in the I/O Configuration Program (IOCP) source deck represent logical channels, control units, or device addresses. Therefore, the IOCP source must be created in such a way that devices in the same logical control unit range are in the same physical control unit range and physical channel path ID (CHPID). Similarly, devices on the same CHPID and in the same physical control unit range must be specified as being in the same logical control unit range when assigning device numbers to them. Tape restart will then be able to check the device numbers obtained from the channel subsystem during system initialization against the logical channel and control unit definitions produced by the IODEV macro.

## Customizing Tape Processing Options

SIP generates keypoint record A (CTKA), which contains bit switches for test mode operation (normal vs. test) and real-time tape operation (normal vs. merge to RTA). The default settings for both of these bits are normal. To change these settings, the assembled keypoint record must be modified via patching.

The following tape options are allowed by modifying the field that maps to CPMOPT in SKCTKA. The proper field modification is listed by each option.

- B'xxxxxxx1'** For merging the output of both the RTA and RTL tapes into a single tape, the RTA tape. If storage dumps are being recorded on the RTL tape and the RTL tape is dismounted, storage dumps are

temporarily routed to the RTA tape regardless of this bit setting. If this bit is set off, dumps are automatically rerouted to the RTL tape when the RTL tape is remounted.

This option is intended for use by test systems and, if used in a production system, should not be changed once the system is generated.

**B'xxxx1xxx'** For setting logical write protect on devices that are not attached to the tape library dataserver.

**B'xx1xxxxx'** For mounting fixed-length format tapes for input.

The system treats these as undefined-length format tapes.

**Notes:**

1. If any of these options are desired, you must modify the field before a system generation.
2. Any combination of these options can be selected. For example, to select all three of the preceding options, the CPMOPT field would contain B'xx11xxx1'; where 1 selects the option and x can be 0 or 1.

## Customizing Stalled Tape Module Queue Monitoring

You can control whether TPF monitors stalled tape module queues and how many seconds may elapse before a stalled queue is reported through the ZCTKA command. In setting the timeout value, you might start by setting the value to 10 seconds and then determine if that amount of time is adequate for detecting stalled tape module queues on your system. If the timeout value is too low, the system overhead increases. If the timeout value is too high, the stalled tape module queue may not be detected quickly enough to recover before a problem causes a system outage.

## Allocating Additional Working Storage for LIFO Devices

TPF-supported tape units contain buffers that are used to increase I/O performance. Devices with buffers can receive errors that require the records in the control unit to be retrieved and written to a new tape. The process of retrieving the records and writing them to a new tape is referred to as dynamic device reconfiguration (DDR). TPF supports two modes of DDR: first-in-first-out (FIFO) and last-in-first-out (LIFO). See the *TPF Migration Guide: Program Update Tapes* to determine which tape units and licensed internal code are required to support FIFO control unit buffer recovery.

If all tape drives in a system are capable of FIFO recovery, there is no need to allocate any additional working storage for DDR. If there are some tape drives in a system that are not capable of FIFO control unit buffer recovery, it is recommended to allocate at least 1-1/2 times the maximum number of blocks required to recover the control unit buffer (including the SWBs necessary for requeuing the records) for each block type in addition to normal allocation. This will allow for two buffer recoveries to take place, one after the other.

**Note:** For records that are larger than 4 KB in size, buffer recovery will use multiple 4 KB blocks to recover the record.

## Defining Default Categories for the Tape Library Dataserver

TPF supports the tape library dataserver through the ZTPLF command. If the ZTPLF RESERVE and ZTPLF RELEASE commands are not supported for your version of the tape library dataserver, default categories (categories into which



volumes are moved after they have been used by the TPF system) are defined in TAPEQ. One value is assigned for each processor in the complex as shown in the following:

```

Processor 0 - X'1000' tag name DEFLT0R0
Processor 1 - X'1100' tag name DEFLT0R1
Processor 2 - X'1200' tag name DEFLT0R2
Processor 3 - X'1300' tag name DEFLT0R3
Processor 4 - X'1400' tag name DEFLT0R4
Processor 5 - X'1500' tag name DEFLT0R5
Processor 6 - X'1600' tag name DEFLT0R6
Processor 7 - X'1700' tag name DEFLT0R7
Processor 8 - X'1800' tag name DEFLT0R8
Processor 9 - X'1900' tag name DEFLT0R9
Processor 10 - X'1A00' tag name DEFLT0R10
Processor 11 - X'1B00' tag name DEFLT0R11
Processor 12 - X'1C00' tag name DEFLT0R12
Processor 13 - X'1D00' tag name DEFLT0R13
Processor 14 - X'1E00' tag name DEFLT0R14
Processor 15 - X'1F00' tag name DEFLT0R15
Processor 16 - X'2000' tag name DEFLT0R16
Processor 17 - X'2100' tag name DEFLT0R17
Processor 18 - X'2200' tag name DEFLT0R18
Processor 19 - X'2300' tag name DEFLT0R19
Processor 20 - X'2400' tag name DEFLT0R20
Processor 21 - X'2500' tag name DEFLT0R21
Processor 22 - X'2600' tag name DEFLT0R22
Processor 23 - X'2700' tag name DEFLT0R23
Processor 24 - X'2800' tag name DEFLT0R24
Processor 25 - X'2900' tag name DEFLT0R25
Processor 26 - X'2A00' tag name DEFLT0R26
Processor 27 - X'2B00' tag name DEFLT0R27
Processor 28 - X'2C00' tag name DEFLT0R28
Processor 29 - X'2D00' tag name DEFLT0R29
Processor 30 - X'2E00' tag name DEFLT0R30
Processor 31 - X'2F00' tag name DEFLT0R31

```

For tape library dataservers that do not support the ZTPLF RESERVE and ZTPLF RELEASE commands, the CORB segment uses a table, created from the TAPEQ values, to determine the correct default category. Users can change default categories by changing TAPEQ values and reassembling CORB, or by adding code to the CORU user exit.

**Note:** Enter **ZTSTB 2** to display the default category for a device in Tape Status Table – Section 2.

## Defining a Minimum Tape Level for the Tape Library Dataserver

When you issue the ZTPLF FILL command, the tape library dataserver keeps a tape device filled with volumes from a specified FILL category. To prevent you from running out of the FILL category tapes, use CMMNBRV to predefine the minimum number of tapes allowed for the FILL category. (CMMNBRV sets the minimum tape number value in CK1VOLS in CTKA. CK1VOLS is defined in CK1KE.) If the number becomes less than the predefined minimum, the operator is notified of the condition by a system message.

---

## Unit Record Support

Unit record equipment may be used by low-priority batch-oriented entries (such as the generation of reports) which require printers or card readers as input/output devices. Utility programs such as card to tape, tape to printer, and so on, are provided and may be used in conjunction with user written routines. Unit record

support itself is optional and is included/excluded based upon whether or not the user codes a UNITRD macro as part of his SIP input. Unit record support is by processor in a LC complex and by subsystem in a MDBF system.

Entries using unit record equipment tend to hold core blocks for a long period of time relative to the time required by other I/O devices. Thus, if indiscriminately activated, entries utilizing unit record equipment would impact the performance of the entire system. Priority tables in the unit record message editor program permit low priority unit record entries to be activated only when a predefined number of 136-byte storage blocks are available. Higher priority entries may be activated any time, thus the need for operator discretion.

Like general tapes, unit record devices are assigned to ECBs. This ensures consecutive printing and card reading. Unlike general tapes, however, no reserve unit record function is provided to pass control of unit record devices to another ECB for further processing.

The SIP UNITRD macro is used to specify the characteristics and requirements of unit record equipment. The Unit Record Status Table (see data macro UR1ST) contains information reflecting the status of the unit record devices. Six bytes per URST entry are contained in keypoint record B (CTKB). SIP uses the UNITRD macro parameters defining unit record devices to generate this information in macro CK9KC, the CTKB DSECT.

In an LC complex, only one Unit Record Status Table is defined. Since programs requiring unit record devices are invoked by operators, it is the responsibility of the central site operator to ensure that the unit record devices are switched to the desired processor within the complex before invoking a task requiring unit record devices.

The following parameters of the SIP UNITRD (unit record) macro relate to unit record support:

- UNIT (device type)
- ADDR (device address)
- SYMNO (symbolic number)

The Control Program Link Map Table is a control program CSECT which provides a directory of the control program CSECTs loaded to the online system. It is also used online by the same program for printing the control program link map table to the printer. The options are only available to users with unit record support.

Inclusion of unit record support is not required for use of the in-core dump formatter (ICDF) in a test system. Use of this facility is requested by the APRNT and DUMPDEV=TAPE parameters of the CONFIG macro. The same printer, as was specified by the APRNT parameter, may also be specified as a printer in the UNITRD macro.



---

## Database Support

The user has the option to install the HPO feature and to make use of the multiple data base function (MDBF), in which case the database of one application or group of applications can be separated from the database of another via the use of subsystems (SS). Using MDBF within a single subsystem, including the basic subsystem (BSS), the user may also have a logical separation based upon the concept of subsystem users (SSU). MDBF allows the database of any given subsystem to be physically separate from that of any other SS, including the BSS. In the rest of this section of the document the term *system* should be replaced by the MDBF user with the term *subsystem*. Only in those cases where there is any other meaning will the difference be highlighted for MDBF users.

TPF assumes the responsibility for assigning sets of data to physical devices and converting user data references to physical locations. Thus, the application code will not have to change when the physical configuration of the data files is changed. The system will also be able to arrange the data structure on the physical devices to meet system objectives. TPF favors the objective of performance.

Sharing DASD control units between TPF and another operating system (such as MVS) will have a **significant impact** on system performance. Contention between operating systems for control unit access can result in busy conditions that will impair the execution of I/O operations.

The purpose of file storage is to hold programs and data that are requested by entries and then transferred to main storage. The performance of a system mainly is dependent on the number of file storage requests and the time required to transfer them. The number is largely determined by the application design. The request time includes the amount of time taken by the control program instructions, the time required to find the requested block on a physical device, the time in queue for the device, and the transfer time. Since each device can only handle one request at a time, multiple requests for a particular device must be queued.

The transfer and seek time are based on the device characteristics. The queuing time, however, is dependent on the file organization. A first step in reducing the queue for any device can be to design a system that approaches equal queue sizes for each device. The average length of the queue on a device can then be reduced by increasing the number of devices.

Two factors governing file design are the capacity of the files in relation to the data to be contained and the ability of the files to handle requests at a rate consistent with the requirements of system performance. As queuing increases, the data-request time increases and the life of an entry increases; thus, a larger demand is made on working storage. This results in either a reduced message rate or a system failure due to a lack of working storage. The following paragraphs describe file storage support:

File storage is organized for 3 record sizes. These sizes (381-byte, 1055-byte, and 4KB) are common to all file storage media and are consistent with the core block sizes in working storage. The difference in size between these and core blocks is due to core doubleword boundary requirements and format flags, which are only maintained in memory, and need not be concerned with the storage media.

The file storage media supported in terms of the basic unit are listed in *TPF Migration Guide: Program Update Tapes*.

The organization of a file storage system is intended to distribute the records associated with a set of data, called a record type, over physical file storage to reduce queuing time at the devices. Thus, when an application accesses successive records within a record type, each record is obtained from a different physical component.

A record type contains data records that are related by application or by usage. Different record types may be files of programs, inventory records, or payroll histories. Within applications or subsystems there may exist many record types. The same record type may exist in different MDBF subsystems, but it is physically a different set of data records.

Traditionally, MVS record types are called logical files and are customarily assigned to one or more storage media units via a vertical allocation. This has an advantage both in isolating particular files for protection and processing, and in allowing an installation to reduce the number of media units when the file does not have a requirement for continuous availability. This latter advantage is lost in a real-time environment when all logical files must be available at all times. The main disadvantage of such an organization arises when the applications on a particular logical file require a high activity. This results in a queuing problem, especially when using DASD devices. Trying to balance logical files on storage media is very difficult and fails when the application logical file profile changes during operation.

TPF minimizes the problem of uneven queuing on these particular drives by distributing the logical files over all the units of the particular device type (See Figure 10). There can be up to four different device types, each of which is made up of like physical DASD devices. These are referred to as *DEVA*, *DEVB*, *DEVC* and *DEVD*. This distribution is mandatory and automatic for all DASD devices, with each type having its own distribution. The organization might be envisioned as a group of disk packs, each identical in format and type of contents. The logical files are layers on the packs, each pack with an equal percentage of the total logical file.

Figure 10 represents four disk drives. Device types cannot be intermixed. Residing on these drives are three logical files or record types (XX, YY, and ZZ). Each logical file shares an equal portion of each physical device.

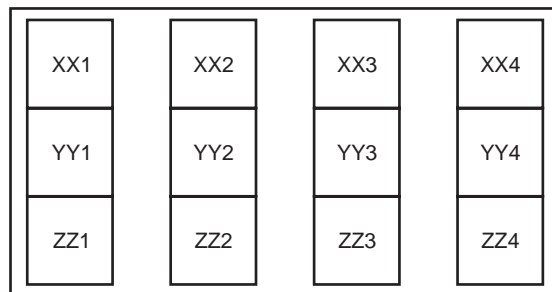


Figure 10. Logical/Physical Disk Device Relationship

The distribution technique in organizing fixed file storage is to place the first record of a logical file on the first physical device, the second record on the second device, and so on to the Nth device, after which the devices are repeated (that is, logical records 1 and N + 1 are on the first physical device). (See Figure 11 on page 59.)

The purpose of this organization is to allow a larger number of concurrent accesses to any particular logical file than would otherwise be possible, thereby reducing the

chance of excessive queuing. It has an additional advantage in that it frees the application design from many of the physical device performance considerations.

Each of the physical device types has its own organization. Logical files are allocated on each type according to the rules of that type. Logical files can be split, with each portion placed on a different device type (see Figure 12). Note that all records allocated to a particular device type are contiguous.

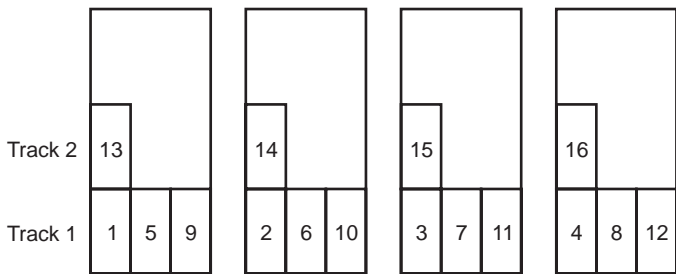


Figure 11. Record Allocation. This is an expansion of Figure 10.

As can be seen, consecutive records are placed on consecutive devices

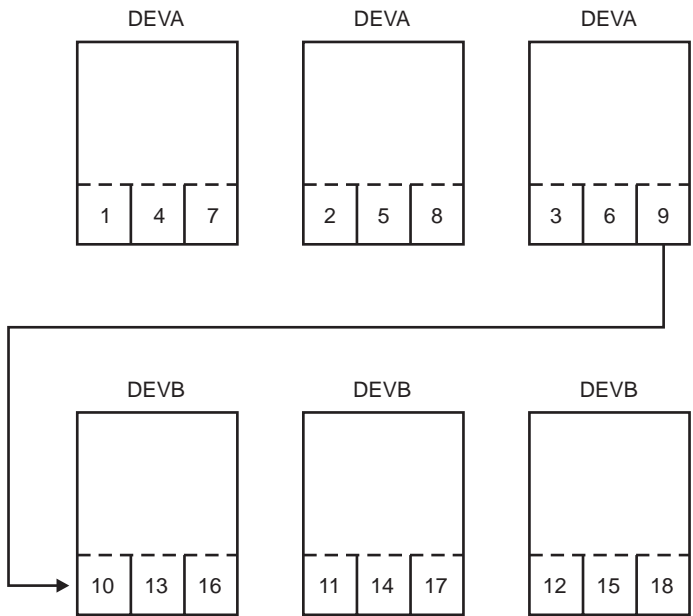


Figure 12. Multiple Disk Device Type Record Organization

The type of file addresses used by an application program is called the File Address Reference Format (FARF). The techniques used by the application to obtain addresses relate to the record definition on the files. Records on file storage are classified as either fixed or pool records based on how records were requested.

The fixed record area is an area of file storage that contains records whose symbolic addresses can be calculated using a record type and an ordinal number. The record type identifies a set of data within the fixed record area, and the ordinal number identifies a specific record within the record type.

As an example, consider an inventory file for an airline. A record type for flights can be identified and a record can be set aside for each possible flight number on each

date for which inventory is kept. Using the flight number and date, the sequence of the appropriate record can be determined. Today's flight 300 would be on the three hundredth inventory record, tomorrow's on the  $300 + n$ th record, where  $n$  is the maximum value of a flight number. Obviously this technique is wasteful if the number of records set up far exceeds the requirements of usage. When fixed records are set up for logical files, file storage is allocated and maintained for the exclusive use of each possible record in that logical file.

A different method of obtaining an address must be employed when either the number of records used is far less than those required on a fixed record scheme or the address cannot be calculated. (Although it would be possible in a logical file of names to allot a record to every possible name, it would be unreasonable in terms of the file storage required.) To solve this problem, a number of records are set up in file storage whose addresses are managed by the system itself. When an application needs a new record, the system gives it the symbolic address of one of these records in the pool. When the application is through with the record, its address is returned to the system for recycling. TPF ensures that an address is not given to more than one requester prior to its release. The records set up for this purpose are called pool records.

In order to retrieve a pool record, the application must be able to obtain its address. The common technique used by applications is to form data structures that use fixed records as indexes to pool records. For example, the reservation application uses a fixed record to locate passenger name records in the pool. The index is retrieved, using the flight number and date to calculate its symbolic address. The index is scanned for the passenger's name, and, when it is found, the associated pool address is used to locate the passenger name record. This allows the selection of one of a vast number of pool records with a minimum of file accesses.

Duplication is insurance against loss of critical data due to hardware failure. If a record is duplicated, two copies will be updated whenever file storage is updated. Duplicates are kept on separate devices. When requested, either copy can be used, dependent on queuing on the devices and/or the inability to retrieve one of the copies. TPF supports the duplication of any or all logical files in both the fixed and the pool areas.

When using the loosely coupled (LC) facility of the HPO feature, be aware that record duplication presents a unique situation because it requires the use of the external lock facility (XLF). Even if a record is duplicated, if it is retrieved with a hold, only the prime copy is retrieved. The concept of least queuing for either the prime or duplicated copy in a non-LC system is not employed. The reason for this is that the locks for the record, maintained by XLF, are only maintained for the prime record. If this was not the case, one processor could lock the prime record on one CU but another could lock the duplicated copy on another CU. The duplicated copy is only used when there is a failure on either the prime CU or DASD device or for record retrieval without the hold option where least queuing is still used. For the hardware error condition the prime locks are copied from the prime CU to the CU of the duplicated record to ensure that only one processor is allowed to hold a record at any given time.

The rest of this section is subdivided into sections addressing the various database device types and record types. Database device addressing and database allocation are addressed in "Database Device Addressing" on page 79 and "Database Allocation" on page 91 respectively, due to the interdependence of the various device types and record types on the overall database design.

---

## Disk File Storage

Supported disk file storage devices are listed in the *TPF Migration Guide: Program Update Tapes*. Disk file storage devices in a TPF system will typically contain approximately 98 percent of the database volume. The database may be divided into the following areas. Note that the order of the areas in this figure is not necessarily representative of the records on DASD.

Table 3. Database Areas

Database Area	Duplication Status
Permanent Record Area	Not duplicated
Core Image Restart Area (CIMR)	(Note 1)
IPL Area	(Note 1)
Keypoints - 4K Fixed Records	(Note 1)
4K Application Programs, Fixed	(Note 2)
4K Fixed Records	(Note 2)
Large Fixed Records	(Note 2)
Small Fixed Records	(Note 2)
4K Long-Term Pool Records	(Note 2)
4K Long-Term FARF6 Pool Records	(Note 4)
Large Long-Term Pool Records	(Note 2)
Small Long-Term Pool Records	(Note 2)
4K Short-Term Pool Records	(Note 3)
Large Short-Term Pool Records	(Note 3)
Small Short-Term Pool Records	(Note 3)

**Note 1:** Must match device duplication.

**Note 2:** These areas are optional for duplication.

**Note 3:** Short-term pool records are coded as nonduplicated.

**Note 4:** This area must be coded as duplicated.

The primary facility to aid in maximizing the availability of records is record duplication. The amount of duplication is a function of how critical the information is to the system. Also to be considered is the ease of recreating lost data, volume of records of a particular type, and time spent to FIND or FILE duplicated records.

Thus, critical, highly used records and those difficult to recreate should be duplicated. Noncritical and/or infrequently used records and those which are quickly and easily recreated need not be duplicated. Records can be duplicated on file for immediate availability or they can be logged on tape and reloaded to effect recreation. It should be noted that there is a tendency by all users to fully duplicate all system records.

A user may choose any one of the following three alternatives for file layout:

1. Nonduplicated system
  - None of the records are duplicated in the system. Under such circumstances, the optimum file layout can be achieved by grouping the most active records at the center of the DASD.

## 2. Fully duplicated system

- In a fully duplicated file system, all records are duplicated on another module. The primary copy of file is kept on a DASD called the *prime module*, while the duplicate file is placed onto *dupe module*. The term *pack* is used interchangeably with *module*.

## 3. Partially duplicated system

- A user may want to duplicate only certain fixed records and certain pool records. The duplicated records will reside on a different module. Such a decision will be governed by the total file space available, individual amount of space required by each type of record, and the critical nature of the records.

### Notes:

1. Nonduplicated fixed records are found only on the prime packs. The related space on dupe packs is unused.
2. Nonduplicated pool records are spread across all packs in a selectively duplicated system.

Duplication can include all records, no records, or selected areas. TPF uses the method of module-to-module duplication. Module-to-module duplication provides for an optimum seek time by allowing for greater flexibility in positioning records on the module. It is the easiest method to initialize. Module-to-module duplication can be utilized only with configurations which have (including planned growth) an even number of modules.

Module-To-Module Duplication

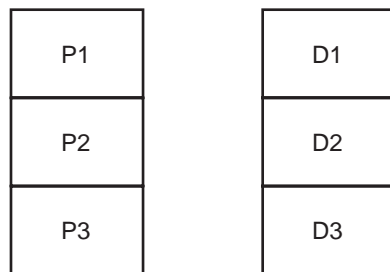


Figure 13. File Duplication Methods

There are two basic principles in minimizing the access time. The first is to spread the access load evenly across all modules and the second involves minimizing the access time for each individual module.

The access load is spread evenly across the modules by allocating an equal portion of each record type on each module. Thus, for an  $n$  module system, each module contains  $1/n$  of the records of a particular type.

The second objective, for example, minimizing the access time for a module, is achieved with the following considerations:

- The most active records should be grouped together at the cylinders where the moving arm is at the center point of full excursion.
- Reading of a duplicated record is achieved by queuing the request either on the primary module or the duplicate module, wherever the queue is smaller.
- Updates of duplicate records affect both the primary and the duplicate record.
- The number of reads exceeds the number of writes on certain types of records.

- The total number of records of each type.

---

## Fixed File Storage

The records which need to be frequently accessed and which may hold pointers to data held in the pool areas, are called fixed-file records. A set of data is identified as a fixed record type when it represents a collection of related records. Within the fixed record type the total number of records must be specified and must be designated as small, large, or 4K.

The identification of a set of data as a fixed record type accomplishes several things:

- The system allocates the fixed-records to physical storage by using a strategy which favors performance.
- The application programmer can associate an order to records independent of the physical location of the records.
- The system can physically move records within the file storage system without requiring a change to the record name within the application code.

The fixed record types required by the applications which run under TPF should be defined at the time of system generation. Contrast this with the batch orientation of data definitions within MVS where data sets are defined with a combination of Job Control Language, source code and supervisory service routines; that is, the complete description of the data set is always deferred until job execution. The definition of all fixed file records at system generation has the advantage of eliminating the overhead of describing the characteristics of a set of data each time a unit of work is to be processed. This forces all the files which are required by the online applications to be physically in place and available, which is simply a characteristic of an online system that runs in real-time and accepts random input.

Each of the records associated with a record type is identified by an ordinal number. Ordinal numbers are meant to indicate the succession of records within a fixed record type. For example, consider a record type, T, and for purpose of exposition let the data in the first 4 records be represented as DATA1, DATA2, DATA3, and DATA4. If a request is made for records 1–4 of record type T, then the system should yield DATA1, DATA2, DATA3, and DATA4 regardless of the physical location of the records. This means that even if the records are not physically adjacent in the physical file storage system, the system should consider these records as the first, second, third and fourth records of record type T.

To accomplish the fixed file organization requires two things:

1. A system table identifying where the various record types are assigned. The creation of this table is the essence of allocation and must consider the physical device characteristics, system configuration, and the allocation strategy. This main storage resident system table has pointers to the origins, called base addresses, of the various fixed record types. This table is known as the file address compute table (FCTB). It is created using the FACE Table Generator (FCTBG).
2. System programs generate a physical address given a fixed record type and ordinal number. The mapping of a fixed record into a physical address is accomplished by a combination of I/O service routines and the file address compute program (FACE). Two versions are provided; the first accepts the record type as a symbolic equate called the record type (FACE), the other accepts character input of the record type (FACS). FACS is the preferred



interface, because it makes the application code independent of the FCTB. The FACE interface may require the application to be reassembled if the value of the symbolic equate changes.

The preceding functions occur at distinctly different times. The allocation occurs once, at system generation, and produces the system table of base addresses. The mapping of a physical file address occurs each time an application needs to obtain the physical location of a fixed record. The mapping programs use the FACE table generated by the FACE table generator.

The system views the entire file space as a repository for holding 4KB, 1055-byte, and 381-byte records. In a given file configuration, the system is capable of naming, through the use of ordinal numbers, some maximum number of records. Clearly the system can have no more fixed record types than this maximum. In practice, the number of fixed record types is much smaller than the total number of records. Consider the representation of the fixed file space of  $n$  records (0 through  $n-1$  ordinal numbers) in Figure 14. The number of record types is given as  $m$ , where  $m$  is less than  $n$ .

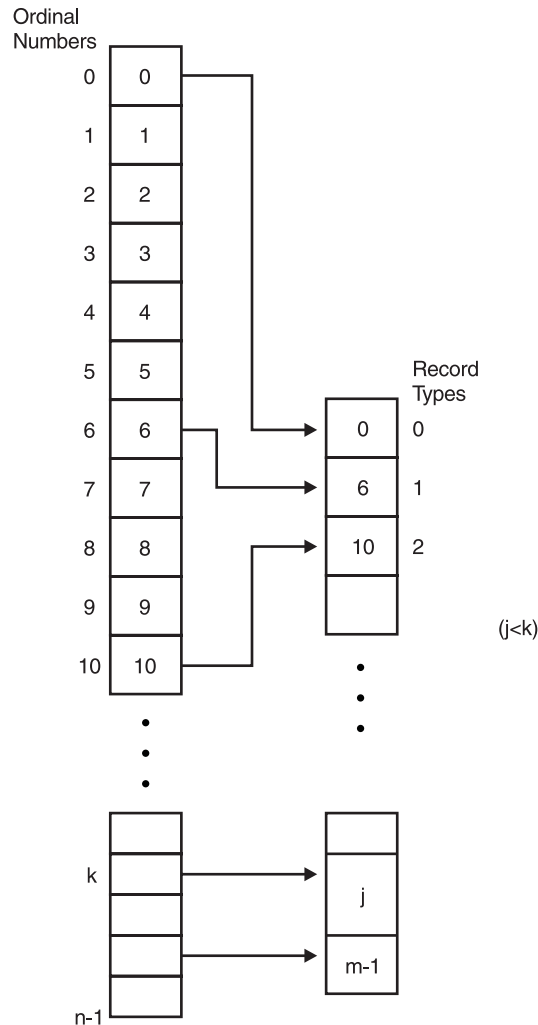


Figure 14. Fixed Record Space

To allocate fixed file space means dividing the entire fixed file space into the subgroups of record types (a record type consists of either all small, all large, or all



4K records). The allocation is accomplished by creating the FACE table (using SIP RAMFIL statements), which associates a beginning ordinal number (called the record type base address) with each record type. Each ordinal base address, after the first one, accounts for the number of records assigned in the preceding fixed record type. For example, in Figure 14 on page 64 record type 1 begins at ordinal number 6 and four records have been allocated.

The next record type (type number 2) begins at ordinal number 10. In general, each record type is associated with some unique ordinal number, k. This unique ordinal number is known as the fixed record type base address. Notice that the second record (ordinal number 1) of record type 1, is at ordinal number 7 of the overall fixed file space in Figure 14 on page 64. It is important to notice that:

- A fixed file area for holding record types consisting of records of the same size and has a set of ordinal numbers (0 through n-1).
- Each fixed record type has a set of ordinal numbers equal to the number of records designated for that record type.

Computing a file address means converting a record type and ordinal number into its file address reference format, or FARF. This is accomplished by looking in the FACE table to find the split that describes the record type and ordinal requested. Fixed file records are loaded from a pilot tape using one of the FACE programs to determine their addresses or they may actually be created in the execution of the online system by an initialization or maintenance program. For more information about TPF DASD addressing, see *TPF Database Reference*.

## RAMFIL Statements for Fixed Record Types

Specifying the characteristics of fixed and pool record types on direct access storage devices is accomplished via the SIP RAMFIL statement. A separate RAMFIL statement is required for each fixed record type, spare record interval, and pool type. The following are the SIP RAMFIL parameters relating to fixed record areas:

<b>RECID</b>	FACE record type
<b>RECNO</b>	Number of records being specified as a group
<b>DUPE</b>	Type of duplication
<b>TYPE</b>	Record size and storage device type
<b>BASE</b>	Starting record address
<b>PRIOR</b>	Priority for split groups
<b>INUSE</b>	Actual vs. spare record types
<b>BAND</b>	Record type band number
<b>USER</b>	Subsystem user owning record (MDBF only option)
<b>UFTI4 UFTI5</b>	UFT/FTI combination for addresses
<b>DUMPALTMODE</b>	A parameter used during conversion of databases
<b>RESTORE</b>	Controls whether record type should be restored or not
<b>UCOMDATA</b>	User record information attached to the FACE table
<b>UFARF3 UFARF4 UFARF5</b>	User FARF data attached to individual splits

<b>EQU</b>	Record type equate value associated with a record type
<b>OWNER</b>	Owning SSU, processor and I-stream for a record type.

The sequence of the RAMFIL statements determines the sequence of the record groups on file storage. Record types may be divided across multiple device types or placed on noncontiguous locations of the same device type. A separate RAMFIL statement is required for each such division across device types. The PRIOR parameter is used to specify the sequence priority.

The INUSE parameter of the RAMFIL statement permits the specifying of spare record types for later use.

The UFTIn parameter specifies which UFT/FTI combination the addresses are allocated from. The available 4-gigabyte address space should be thought of as sliced into 64 sections. Each section is called a universal format type (UFT). Each UFT is further divided into groups represented by unique format type indicators (FTIs) with the same UFT. One group may be FARF4 while another group may be FARF5. The group may vary in size. There may be 10 FARF4 records followed by 20 FARF5 records, and so forth for the remainder of the UFT slice. The FTI indicator for a particular record in a particular UFT has a length associated with it. This length is used to determine the granularity in which records can be defined. The ordinal included on the end of the address specifies the offset within this group of records where a particular record resides. The UFTIn parameter provides the means for specifying what addressing format should be used to interpret records found within the record types being defined by the RAMFIL statement.

The USER parameter specifies which subsystem users, processors, and I-streams have access to particular record types. By default any subsystem user, processor and I-stream can access any record type. Using the USER parameter access to a particular record type can be limited to specific subsystem users, processors or I-streams.

For example, you may want to monitor I-streams for a certain reason. You can specify groups of record types related to each I-stream that are I-stream unique: USER=(\*,\*,1) for group 1 accessible from I-stream 1 only, USER=(\*,\*,2) for group 2 accessible from I-stream 2 only, and so forth.

Alternatively, you might want to specify a group of record types that are only accessible from a certain subsystem user: USER=(SSU1,\*,\*). This specification says that the records are accessible to subsystem user SSU1 from any I-stream or any processor.

The combination of these three elements can be thought of as forming a three-dimensional solid or matrix. For example:

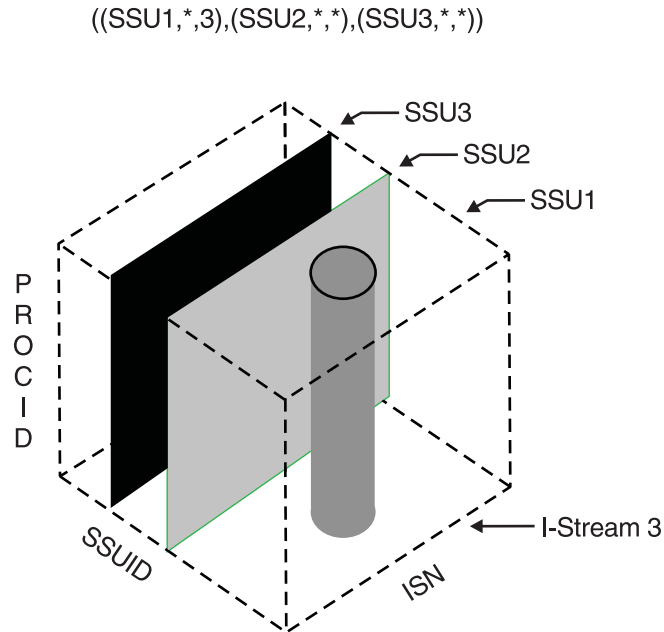


Figure 15. Three Dimensional Example of Record Uniqueness

describes the relationship between three record types: one is accessible only from subsystem user 2 but on any processor or I-stream, another similarly but only from subsystem user 3, and the third only from subsystem user 1 and only on I-stream 3 but from any processor. The \* indicates a given combination applies to **any** processor, subsystem user, or I-stream, depending on the field where \* appears.

**Note:** While the combination of these elements, subsystem users, processors, and I-streams can be thought of as forming a three-dimensional matrix, the FACE table actually uses a combination of one-, two-, and three-dimensional matrixes to implement record uniqueness. The FACE table generator (FCTBG) creates a one-, two-, or three-dimensional matrix for each record type depending on how the USER parameter is defined on the RAMFIL macro. When possible, one- and two-dimensional matrixes are used instead of three-dimensional matrixes to save space and reduce the total size of the FACE table.

Consider the example:

USER=((SSUA,D,\*),(SSUB,C,2),(SSUG,\*,\*),(SSUF,\*,6))

shows a record type being specified as being accessible in four different places:

1. Subsystem User A on processor D and any I-stream
2. Subsystem User B on processor C and I-stream 2
3. Subsystem User G on any processor and any I-stream
4. Subsystem User F on any processor on I-stream 6.

USER parameters specified using a single \* can be thought of as vectors. When there are two \* characters they can be thought to form a plane.

Specifications using \* should be thought of as *implicit*, while specifications specifying values other than \* are *explicit*. Explicit specifications always override implicit ones.

---

## Pool File Storage

Pool file record addresses are obtained by issuing a get file storage (GFS) type macro. The application program may then store data in the record referenced by the given file address for whatever period of time is appropriate. A file address for a pool record is returned to the system through the use of a release file storage (RFS) type macro.

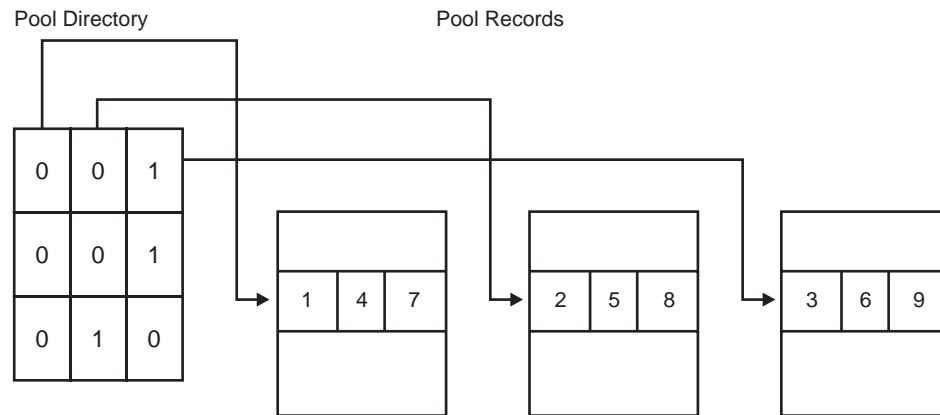
Pool file storage is managed through the use of pool directories built by offline programs and assigned to online system residency by special programs invoked in an initial system restart. There are several types of file pools based upon the attributes of:

- Longevity (long or short term)
- Record size (381, 1055, or 4095 bytes)
- Duplication (yes or no)
- File address format.

The various types of pools may be considered as system record types. Each type of pool is allocated some maximum number of records (usually a very large number). The base addresses of the various pool types are identified in the SYCON macro which is generated by FCTBG, and are held in keypoints during online execution for the use of the get file storage macros. The space required for the pools is accounted for in the FACE table.

Rather than maintain a unique space consuming a file address for each record, the system maintains a bit indicator in a pool directory. Each record within a pool type is indicated with a bit (for example, 0 or 1). The relative position of a bit within a directory determines an ordinal number within a pool type, shown in Figure 16 on page 69. Upon a request for file storage, the GFS macro interpreter scans the directory for a 1. Upon a match, the relative position number within the directory is used as an ordinal number and the bit is switched to zero. A release file storage macro interpreter switches the availability bit of a short term pool record back to a one. In the case of a release, a FARF address must be converted into a relative bit position in a directory.

FARF address references in the pool area are generated by the get file storage facility. The resulting address is passed to the requesting program. This is the address to be referenced as a parameter of the FIND and FILE macro requests. The FIND and FILE macro interpreters perform the additional translation necessary to obtain the physical module number by using the FACE Table. Therefore, pool addresses, as well as fixed file addresses are independent of a change to online disk pack hardware addresses (for example, the system operator may have a need to bind a symbolic module number to different hardware addresses during online system activity).



Scan Left to Right  
0 = Not Available  
1 = Available

3 Module System (with records allocated across the modules)

Figure 16. Sample File Pool Directory

File pool records are classified according to the attributes of size, duplication, and length of intended use. The 10 basic pool types are:

- Small, long-term record pool (SLT)
- Small, short-term record pool (SST)
- Small, long-term duplicate record pool (SDP)
- Large, long-term record pool (LLT)
- Large, short-term record pool (LST)
- Large, long-term duplicate record pool (LDP)
- 4K, long-term record pool (4LT)
- 4K, short-term record pool (4ST)
- 4K, long-term duplicate record pool (4DP)
- 4K, long-term duplicate FARF6 record pool (4D6)

All pool records are duplicated in a completely duplicated file system. Only long-term pool records may be duplicated in a partially duplicated file system.

Pool records are not loaded like fixed records since the pool records are only used during execution of the online system. However, the directories which control the use of this area are loaded in the fixed file area by the online directory generation programs.

The recommended way to acquire pool addresses is by use of the GETFC macro, supplying the desired pool record ID only. The system then determines from the record ID attribute table (RIAT) the pool type and size (small, large, or 4K) required. The next available pool record of the requested type is assigned to the operational program. The program must then record the address of the record in order to later find the data it has inserted in it. Duplicates are obtained in the same manner as duplicated fixed file data records. (See *TPF General Macros* for more information about the GETFC macro.)

After fixed file requirements and their location on file have been determined, the remaining file space is available for assignment of file pool storage.

The considerations that should be taken into account in the assignment of file area for pool usage are:

- Requirements for a given pool can only be estimated at system initialization time. It is suggested that not all available pool space be allocated at system initialization, but some area be held in reserve for later addition to a given pool as dictated by actual needs.
- A particular pool type does not have to consist of only one contiguous interval on disks. Pool segments are supported in which different areas of disks are tied together as one pool through the directory scheme.

## Ratio Dispensing

The file pool ratio dispensing facility is provided to dispense addresses according to predetermined ratios for any basic pool type using several sections of the pool (for example, SST pool sections). When a pool section is selected, its ratio factor (that is, a number of addresses) is used to determine how many addresses will be dispensed from that section before selecting the next scheduled ratio. This facility thus allows the dispensing of addresses to be spread across several different device types.

A file pool storage record address may be obtained by executing one of the GFS macros for a large (LLT, LST, LDP), small (SLT, SST, SDP), or 4K (4LT, 4ST, 4DP, 4D6) record. The 10 basic file pools may be split across more than 1 (maximum of 4) storage device types. Each split is called a pool section. See *TPF General Macros* for a description of the GETLC, GETSC, and GETFC macros.

File pool ratio dispensing is an optional function selected by the operation of the online system function ZGFSP.

1. When the ratio dispensing option is selected, the user initializes or modifies the ratio dispensing schedule in CTK9 (see data macro CY5GT) online by the ZGFSP command.
2. When ratio dispensing is implemented, upon each request for a file pool address, the ratio dispensing schedule is interrogated. This interrogation consists of first checking the currently selected pool section's ratio counter value. If the specified number of addresses have already been dispensed for that section, the next pool section specified in the schedule is selected. The ratio counter value for that section is then interrogated in the same way, and so on.

For each pool type there are ten sets; each set consists of 2 bytes. In each set, the first byte (a binary counter) contains the number of addresses (up to 100) to be dispensed before selecting the next set. The second byte contains the pool record code check (RCC). Unused sets should be set to zero. Only those devices specified by the RCC in the sets will participate in the ratio dispensing. The schedule is updated as a push-up list with the selected item placed at the bottom of the schedule. This results in a continuous rotation of the dispensing schedule.

The RCC is defined in the CZ1GF macro for various device type basic pool sections. The assignments for DEVA pool sections are listed in Table 4.

Table 4. DEVA Pool Section Assignments

#SLTA EQU 4	Small, long-term
#SSTA EQU 8	Small, short-term
#SDPA EQU 12	Small, duplicated

Table 4. DEVA Pool Section Assignments (continued)

#LLTA EQU 16	Large, long-term
#LSTA EQU 20	Large, short-term
#LDPA EQU 24	Large, duplicated
#4LTA EQU 28	4K, long-term
#4STA EQU 32	4K, short-term
#4DPA EQU 36	4K, duplicated
#4D6A EQU 148	4K, long-term duplicated FARF6

The CZ1GF macro also contains similar equates for DEVB, DEVC and DEVD when the last character of the label is the device type.

## Fallback

When a depleted pool section is selected for address dispensing, and an alternate compatible pool section is used, a fallback situation is said to exist.

Selection of alternate pool sections for fallback processing is done based upon either a user-specified (primary), or a predefined default (secondary) fallback schedule. Secondary fallback is predefined by system design and system initialization.

### 1. Primary fallback

- You have the choice to implement or not implement primary pool fallback. To implement the primary fallback, initialize or modify the primary fallback schedule online by entering the ZGFSP command.
- You can specify a minimum of two or a maximum of six various pool sections (device types) as fallback devices in the online environment by using a command, for example, ZGFSP FLB SLT DEVA-DEVB.

### 2. Secondary fallback

- Secondary fallback is determined by the Pool Cycle-Up Program for short-term pool sections only. You cannot set the secondary fallback pool section. The assignment of a secondary fallback pool section is based on device types and pool type priorities. Devices DEVA, DEVB, DEVC, and DEVD are scanned, in this sequence, first for an available long-term pool section and then for a long-term duplicate pool section according to record size (for example, large or small).

The get/return file pool program (CCSONP CSECT) controls fallback pool selection for depleted pool sections. The Pool Cycle-Up Program initializes file pool fallback schedules as described previously.

If no fallback schedule has been specified or if all pool sections of a fallback schedule are depleted, system error number 000011 occurs unless the schedule pertains to a short-term basic pool type. For this latter case, long-term or duplicate pool sections are designated in the Pool Management Global Table by the Pool Cycle-Up Program as secondary short-term fallback schedules.

The fallback schedules in the Pool Descriptor Record contain record code check (RCC) characters which are assigned uniquely to each pool section. These equates are converted by the Pool Cycle-Up Program to indices, which are then used to select the respective pool sections for fallback.



Each pool type schedule consists of 6 bytes showing the order (left-to-right) in which to fall back to another pool section when all addresses of the pool section have been depleted.

Each byte contains the RCC for the device's pool type. The first byte in the fallback schedule must contain the RCC for the basic device type. Any unused bytes must be set to zeros (null entry). The last valid RCC, after depletion, will cause fallback to the first RCC specified in the schedule.

Any long-term pools not specified in the primary schedule will not have fallback capability. Short-term pools always automatically fallback to equivalent long-term pools when there are no more short-term addresses and no primary fallback schedule is specified. However, when primary fallback schedules are implemented, only short-term pools should be specified for fallback of a short-term pool.

In the coding of the fallback schedules, the null entries have no effect on efficiency, since the fallback schedules are used only when a pool section is depleted. Normally, this will not occur very often.

Specifying the characteristics of pool storage areas on direct access storage devices is accomplished via the SIP RAMFIL statement. A separate RAMFIL statement is required for each pool interval. The sequence of the RAMFIL statements determines the sequence of the record groups on file storage.

The following are the SIP RAMFIL parameters relating to pool intervals:

<b>RECID</b>	Pool
<b>RECNO</b>	Number of records
<b>DUPE</b>	Type of duplication
<b>TYPE</b>	Record size and storage device type
<b>BASE</b>	Starting record address
<b>POLID</b>	Long-term pools or short-term pools
<b>PSEUDO</b>	Number of pseudo modules—allows expansion without pool reorganization
<b>PSON</b>	First pool database ordinal number used for pool segment
<b>UFTI4 UFTI5 UFTI6</b>	UFT/FTI combinations for address allocation
<b>OVERRIDE</b>	Allow RAMFIL statements coded out of PSON order
<b>DUMPALTMODE</b>	Exposes FARF addresses for migration
<b>LOMOD</b>	Starting relative module number for pool segment
<b>NOMOD</b>	Number of modules to spread pool segment over, starting at LOMOD
<b>UCOMDATA</b>	User record information attached to the FACE table
<b>UFARF3 UFARF4 UFARF5 UFARF6</b>	User FARFx data attached to individual splits.
<b>DEACTIVATE</b>	Used to indicate a pool extent to be deactivated.



The SIP ONLFIL macro parameter NAMDEVx is used to change the four-character DASD device name (3350, 3380, 3390) which is used by the pool programs to any four-character name specified by the user. Renaming is required when two device types are identical (DEVICEA = 3380, DEVICEB = 3380) and may optionally be used to change the standard names (3350, 3380, and so on).

File pools residing on DASD devices have their options hard coded in keypoint record 9 (CTK9). After system generation and cycle up, these options may be displayed and altered via the miscellaneous file pool functions system operator command ZGFSP.

Directory records are created for file pools by the pool maintenance program.

Pool directory records reside in the fixed file area of the database. The user must provide SIP RAMFIL statements to allocate file space for these records. Pool directories require record type #SONRI (see data macro CY3DR).

**Note:** When determining the minimum number of pool directories to define, you must consider the number of processors and the set size for your environment.

Use the following formula to help determine the minimum number of pool directories for each pool section:

$$(2 \times \text{set size} \times \text{number of processors}) + 1$$

Users who are required to update one or more of the equate macros in order to generate their system properly should be aware of two cases of when and how to do so. For macros like CZ1GF, which are **not** generated by SIP, the macro should be updated/modified prior to the execution of SIP stage I. On the other hand for macros like SYCON, which **is** generated by the FACE table generator (FCTBG) the user should allow FCTBG to actually create its version of the macro and then the user should update/modify that version of the macro. It is very important that this updating take place prior to the assembly of any programs which call/use the macro in SIP jobs. The alternative is to allow SIP to complete Stage II and then to do the updates/modifications and then to reassemble and relink the affected modules.

---

## Keypoint Records

The control program keypoints are a specific set of critical system records located at fixed locations in the online file storage fixed file area. The keypoints use a 4K record size. A copy of each keypoint record is kept on each of the online modules. Keypoint records are updated on file by a system program in the online management of resources and error recovery. These records represent the **current** status of the system and are fundamental to system generation, system restart and the online operation of the system. Many of the tables contained in the various keypoint records are loaded into the main storage control program records and table area by the initializer program. For additional information on system keypoints and the updating of keypoints, see *TPF Concepts and Structures*.

SIP creates if necessary, and assembles all the control program keypoint records. Many of the SIP macro parameters provide input to the various keypoints.

The system keypoints are located in the fixed file area defined by the FACE record type #KEYPT, and space for these records is allocated via the RAMFIL statement. #KEYPT is a fixed file record and can be placed on DEVA. Another record type related to #KEYPT is #CTKX. CTKX contains the ordinal numbers of every system

keypoint within #KEYPT. This table of keypoint addresses is critical for error recovery and system restart. The user is free to choose which DASD address is used for its location on DEVA. It is ignored if placed on other devices. #KEYPT and #CTKX must have the same duplication as the device duplication.

One keypoint staging area corresponds to each TPF image. The keypoint staging areas are defined by the FACE record type #KSAX, where x is the image number. Keypoints are loaded to an image by writing them to the staging area, where they can then be moved to the physical keypoints (#KEYPT) by issuing the ZIMAG MOVE command. Moving keypoints causes the system keypoints currently in the working keypoint area to be moved into the keypoint backup area. There is 1 keypoint backup area; it is maintained in fixed record area #KBA. Keypoints can be selectively fallen back from this area by entering the ZIMAG KEYPT RESTORE command.

The HPO feature may affect the number of copies which are maintained for a particular keypoint. LC defines keypoints as shared or processor unique. This allows each processor in an LC complex to view a resource, as defined by a keypoint, as either its own or as shared with the other processors in the complex. For example, all information about the status of the shared DASD is located in a shared keypoint.

MDBF defines keypoints as shared or subsystem unique. This allows each subsystem to view a resource defined by a keypoint as its own or as shared with the other subsystems. The MDBF shared keypoints are located in the basic subsystem (BSS). For example, all of the keypoints concerning communications are shared.

It should also be noted that in a loosely coupled complex each processor performs the keypoint copy function independently and it takes into account processor-unique keypoints and shared keypoints described above.

The keypoint copy function is also activated for each subsystem of a MDBF user sequentially and it takes into account subsystem-unique keypoints and shared keypoints. The restart area and the keypoints are applicable for each subsystem and as a result there is an area for each subsystem located on its independent database. A subsystem, other than the BSS, may not be IPLed until the BSS is IPLed since certain records only exist within the core image area of the data base of the BSS. In addition, an MDBF user has the option at generation time to provide the ability to also keypoint the fallback area on the prime module, if desired.

A complete list of the control program keypoint records follows. This summary includes a brief description of the contents of these records. Note that the title of the keypoint (for example, CTKA) is not the same as the data macro corresponding to the keypoint (CK1KE). Data macros declare the names of the fields within the keypoints. They reside in the released control program macro library indicated. Whether keypoints are unique for loosely coupled or MDBF systems is also indicated.

Table 5. 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 5. 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.

Use the following method to determine the number of #KEYPT records needed by each device type in your configuration. This method shows you how to automate the calculation using SIP.

SIP calculates the number of records required for each device type.

1. Code all of your SIP stage I.
2. Code #KEYPT with any number of records.
3. Run the FACE table generator.
4. If the number of #KEYPT records is too small, the FACE table generator produces an error message that tells how many records are needed.

---

## General Data Sets

General data sets are usually related to some offline processing. Data is either produced online (for example, management reports) to be processed by offline (MVS) programs; or data is produced offline for online processing. A general data set provides a data interface between the offline and online system components.

A TPF general data set (a disk module) is directly related to the meaning of an MVS data set. The records of a data set are allocated sequentially within the same module (*volume* in MVS). The file is processed online by using the FIND and FILE macro requests. A special macro, file data chain transfer (FDCT), permits the processing of records that are not restricted to the standard TPF file record sizes. A file with standard record sizes is processed online by using the TPF standard FIND and FILE macro requests. The file referencing is slightly different from the procedures used to access fixed record types and pool records.

Two distinct general data set macros are provided to allow an ECB controlled program to access records within a general data set:

<b>GDSNC</b>	A general data set name macro is used to associate a data set name with a unique entry. (This is functionally similar to an MVS OPEN macro).
<b>GDSRC</b>	A general data set record macro is used to access a specific record within the data set named by the GDSNC macro. (This corresponds to the use of FACE for obtaining a normal fixed or pool file address.)

A command is provided to allow a system operator to mount and dismount volumes associated with a data set. The TPF general data set volumes are formatted by offline MVS utilities. The formatter program produces a header called a volume label. Volume labels are used to identify MVS compatible data sets. See the *TPF Database Reference* for more information about general data set support.

The SIP RAM macro parameter GSON is used to specify the number of slots (spindles not required for online modules) in the DASD file status table. General data sets are mounted on these spare spindles. The spare slots in the file status tables are used to control them.

All general data set control records are obtained from fixed file records accessed through the FACE program. The SIP RAMFIL macro is used to allocate these records. They are obtained and returned as needed and may be used at any point in time for any of the data set control records. Therefore, the total number of data

set records required will vary with the number of data sets to be mounted, the DASD device type and the number of volumes within a data set, and the number of loosely coupled processors.

Data set records must be allocated as 1055-byte records. The size and number of records to be allocated is described in macro DSEQU.

## General Files

TPF general files are sequentially organized sets of data. However, they are not MVS-compatible. There is a limited set of general files required by the TPF system; for example, the general file used to load the pool directories. Therefore, general file support is a required function in TPF.

General file modules are formatted by the TPF formatter (381-byte, 1055-byte, or 4KB records only), and may contain up to 3 data sets per module but with only a single extent per data set. The data set ID number is a unique identification which appears in the volume label. It is a 2-digit number starting from 00 to a maximum of 59. If a data set requires more than one module (overflow) the next sequential data set number will be assigned. A maximum of 60 data sets may be online at any one time.

General files are usually created offline using the *execute channel program* (EXCP) facility of MVS. They may also be created online by an application program.

The SIP RAM macro parameters relating to general files are GFENS and GFMOD. Refer to "Database Device Addressing" on page 79 for an example of these parameters.

The worksheet in Table 6 is for recording general files that are defined to the system. The first 2 are predefined as follows:

- Data Set 2 - File recoup program
- Data Set 3 - Pool maintenance

The file recoup program requires Data Set 2 on a DASD general file. See *TPF Database Reference* for more information.

Volume serial numbers for general files may have any 6 alphanumeric characters, except all Fs.

Table 6. General File Worksheet

Application File Name	Data Set ID Number	Volume Serial Number of Pack
	0	
	1	
File Recoup	2	
Pool Maintenance	3	
	n	

SIP provides the GENFIL macro to define general files to the system. GENFIL parameters are used to build the General File Module Table (see data macro GENFD) in program segment CVZD. Defining general files to an operational system is accomplished by adding entries to program segment CVZD, reassembling CVZD, and then loading CVZD to the online system. In order for this to work, the initial IPL

switch in CTKB (CK9IPLR) must be set to X'00', and CTKB loaded to the online system. This allows the General File Module Table to be reinitialized.

The following GENFIL macro examples define general file data sets 2 through 5:

```
* FILE RECOUP PROGRAM
  GENFIL DS=2,GSIZE=L,CYL=60,TRK=1,DEV=3350
* POOL MAINTENANCE
  GENFIL DS=3,GSIZE=L,CYL=0,TRK=1,DEV=3350
```

The loader general file does not require a GENFIL macro. Because it is required before the online system is fully operational, it is handled separately from the normal general file support.

Parameters provided in the above GENFIL macros are examples.

---

## Database Device Capacity

The SIP IODEV macro is used to specify the characteristics and system requirements for input/output devices, including DASD. Table 7 shows the data capacities of DASD supported by TPF.

*Table 7. Direct Access Storage Device Capacities*

Device Type	Cylinders/ Module	Tracks/ Cylinder	Number of 1055 records/track	Number of 381 records/track	Number of 4K records/track
3350	555	30	15	34	4
3375	959	12	25	46	8
3380	885 * 1770 ** 2655 ***	15	30	53	10
3390	1113 * 2226 ** 3339 *** 10017	15	33	55	12
9345	1440 * 2156 **	15	27	47	10

\* Single Capacity  
\*\* Double Capacity  
\*\*\* Triple Capacity

The 3390 capacities listed above are for native mode 3390s. For 3390s in 3380 emulation mode, use the 3390 cylinders/module; for the records/track, use the same values as the 3380s. The cylinders/module and tracks/cylinder values are unaffected by the mode.

---

## Database Device Addressing

A two-byte symbolic device address is required to uniquely address a given DASD device. The 2 high-ordered hex digits represent the logical channel number of the DASD. The 2 low-ordered hex digits represent the logical control unit, the string address, and finally the logical device. The maximum number of DASD devices allowed per control unit is 64, but it is also possible to define DASD control units that service 32 or 16 DASD devices. In any case you must code the number of devices that will be attached to each controller on the SIP CONFIG DCUSV

parameter. Three examples follow. Figure 17 shows how to interpret the address when a controller services 32 devices. Figure 18 on page 81 is similar, but is for a 16-device controller and Figure 19 on page 82 is for a 64-device controller.

1. In order to uniquely address each of the 32 (CONFIG (DCUSV=32)) drives on a given controller, the 2 low-ordered hex digits are divided as follows:
  - Logical control unit address (bits 0–2)
  - String address (bits 3–4)
  - Logical device address (bits 5–7).
2. The control unit address must start with an even hex digit in the third position of the 4-digit device address for the first 16 drives, and continue through the next higher odd hex digit in the third position for the next 16 drives, for example, 0140–014F and 0150–015F.

Figure 17 shows how the DASD address 03A7 is interpreted.

Unit Address	0	3	A	7
(Binary)	0 0 0 0	0 0 1 1	1 0 1 0	0 1 1 1
	← Channel →		Control Unit	String → Device

Unit Address	String Address	For Devices
03A0-03A7	00	0-7
03A8-03AF	01	8-15
03B0-03B7	10	16-23
03B8-03BF	11	24-31

where:

3 is the logical channel address

A is the logical control unit address for the lower 16 (0-15) drives

B is the logical control unit address for the upper 16 (16-31) drives.

*Figure 17. Example of a 32-Device Controller*

In order to define data paths for 32 drives, four IODEV macros must be coded. The following IODEV macro statements define controller 5 on channel 03 with 32 devices:

```
IODEV IOADR=03A0,DVTYP=DASD
IODEV IOADR=03B0,DVTYP=DASD
IODEV IOADR=03A8,DVTYP=DASD
IODEV IOADR=03B8,DVTYP=DASD
```

where:

3 is the logical channel address

A is one logical control unit address

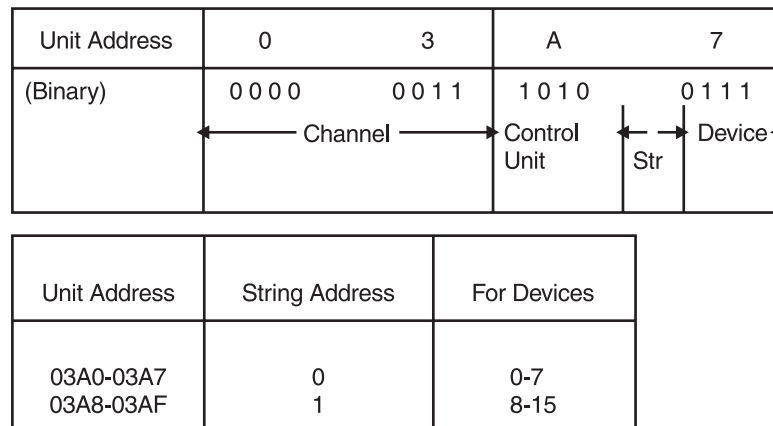
B is same as A, but for devices 16-31.



**Notes:**

1. If the system had been generated to support only 16 drives per control unit (that is, DCUSV=16 in the SIP CONFIG macro), then all CUs would have to be configured so that the CU address is determined by bits 0–3:
  - Logical Control Unit Address (bits 0–3)
  - String Address (bit 4)
  - Logical Device Address (bits 5–7)
2. The logical control unit address may start with any hex digit in the third position of the 4-digit device address. The next higher hex digit in the third position will define another control unit. Notice that this differs from the 32 device example.

Figure 18 shows how the DASD address 03A7 is interpreted.



where:

- 3 is the logical channel address
- A is the logical control unit address.

*Figure 18. Example of a 16-Device Controller*

In order to define data paths for 16 drives, 2 IODEV macros must be coded. The following IODEV macro statements define controller A on channel 03 with 16 devices:

```
IODEV IOADR=03A0,DVTYP=DASD
IODEV IOADR=03A8,DVTYP=DASD
```

where:

- 3 is the logical channel address
  - A is the sole logical control unit address.
1. In order to uniquely address each of the 64 (CONFIG (DCUSV=64)) drives on a given controller, the 2 low-ordered hex digits are divided as follows:
    - Logical control unit address (bits 0–1)
    - String address (bits 2–4)
    - Logical device address (bits 5–7).

Figure 19 on page 82 shows how the DASD address 03A7 is interpreted.

Unit Address	0	3	A	7
(Binary)	0 0 0 0	0 0 1 1	1 0 1 0	0 1 1 1
	← Channel →		Ctrl Unit	← String → Device

Unit Address	String Address	For Devices
0380-0387	000	0-7
0388-038F	001	8-15
0390-0397	010	16-23
0398-039F	011	24-31
03A0-03A7	100	32-39
03A8-03AF	101	40-47
03B0-03B7	110	48-55
03B8-03BF	111	56-63

where:

3 is the logical channel address

8 is the logical control unit address for the first 16 (0-15) drives

9 is the logical control unit address for the next 16 (16-31) drives

A is the logical control unit address for the next 16 (32-47) drives

B is the logical control unit address for the last 16 (48-63) drives.

Figure 19. Example of a 64-Device Controller

In order to define data paths for 64 drives, 8 IODEV macros must be coded. The following IODEV macro statements define controller 8, 9, A and B on channel 03 with 64 devices:

```
IODEV IOADR=0380,DVTYP=DASD
IODEV IOADR=0388,DVTYP=DASD
IODEV IOADR=0390,DVTYP=DASD
IODEV IOADR=0398,DVTYP=DASD
IODEV IOADR=03A0,DVTYP=DASD
IODEV IOADR=03A8,DVTYP=DASD
IODEV IOADR=03B0,DVTYP=DASD
IODEV IOADR=03B8,DVTYP=DASD
```

where:

3 is the logical channel address

8,9,A and B are the logical control unit addresses.

SIP provides the user with a number of macros to define the following online modules and data sets:

- Online disk modules (DASD device types)
- General files
- General data sets

The SIP example in Figure 20 consists of the macros and macro parameters required to define the following configuration:

- Two online 3390s with 6 spares

- Two online 3380s with 6 spares
- Five DASD general file modules
- Two general data sets.

SIP Input Macros	Symbolic Module Number
IODEV IOADR=02A0,DVTYP=DASD	
IODEV IOADR=03A0,DVTYP=DASD	
RAM GFMOD=20,	
NEWGF=YES,	20-24
GFENS=5, GF SLOTS	25
GSON=2	43-48
ONLFIL GDS SLOTS	49-50
DEVICEA=3390,PERMA=2,	26,27
VOLNOA=110,EXTRA=6,	28-33
DEVICEB=3380,PERMB=2	34,35
VOLNOB=2120,EXTRB=6,	36-41
VSNCHAR=TP	

**Note:** Only those SIP parameters related to device allocation are included in this example.

*Figure 20. Sample SIP Parameters for Online Modules*

TPF internally controls the various device types with a set of file status tables. Figure 21 on page 84 lists the 3 tables and describes their organization. In general, these tables include slots for the various file devices supported by the system being generated plus a copy module slot. A copy module slot is a file status table entry reserved for the exclusive use of the file copy system operator message programs. File copy allows the system operator to copy an entire disk module from 1 online module to another (copy) module. The system addresses these tables via symbolic module numbers. While it is not necessary to understand these tables to generate a system, it will be helpful to understand the concept of symbolic module numbers as it relates to the mapping of application program file references (FIND, FILE, and others) to physical hardware addresses. *TPF Concepts and Structures* provides more information on this subject.

Figure 22 on page 84 provides a sequential list of the symbolic module numbers relating to the SIP example in Figure 20. These same symbolic module numbers are included in Table 9 on page 86.

File Status Table (FSTB)	Symbolic Module Number
RESERVED	00,01
General File Module Table (GFMT) (CVZD) DASD pseudo general file numbers	20-24
Module File Status Table (MFST) General Data Set Support 1 entry per DASD device A online module 1 entry per DASD device A pseudo 1 entry per DASD device B on line module 1 entry per DASD device B pseudo 1 entry per DASD copy module 5 entries per DASD general file spindle 2 entries per DASD general data set spindles	25 26,27 28-33 34-35 36-41 42 43-48 49-50 (<4000)

Figure 21. Module Status Table Layout Example

Symbolic Module Numbers		
00-01	RESERVED	
20-24	General File Data Set Slots	General Files
25	General Data Set Support	
26-27	Online DASD Device A	
28-33	Pseudo DASD Device A	
34-35	Online DASD Device B	
36-41	Pseudo DASD Device B	
42	DASD Copy Module	
43-48	DASD General File Module Slots	
49-50	DASD General Data Set Slots	

Data Set Number	Pseudo Module Number
00-04	20-24

Figure 22. Symbolic Module Numbers Example

While the example given includes a variety of device types, only those devices included in a specific generation will require table entries. The one exception is that symbolic module numbers 00 and 01 are always reserved.

The RAM macro GFMOD parameter is used to specify the symbolic module number for the first general file. SIP provides a default value of 230. The example specifies a starting value of 20. The RAM macro's GFENS parameter specifies 5 entries in the general file module table, which means that a maximum of 5 general file data sets can be defined in the system.

Support is provided for up to 3997 online modules per subsystem. The symbolic module numbers in parentheses in Figure 21 on page 84 indicate the maximum symbolic numbers permitted by the various status tables.

General data set support is specified by the RAM macro's NEWGF=YES parameter in the example in Figure 20 on page 83. This function uses the DASD general data set slots to support the general data set modules.

The worksheet in Table 8 is provided for recording the valid channel, control unit, and device addresses for all online and offline (including any expansion) direct access devices. This data will be used to code the SIP IODEV macros for direct access devices.

*Table 8. Direct Access Device Worksheet*

Channel	Control Unit	Device Address
* 03	A	0
<b>Note:</b> * For example, a 3380 DASD device.		

Table 9 on page 86 provides an example of the standard TPF volume serial number format. The inputs coded in Figure 20 on page 83 generated the example. TPF VSNs must contain exactly 6 symbols of the form AANNNN, where:

AA = 2 alphabetical characters

NNNN = 4 decimal numeric characters.

The AANNNN format is mandatory for all online modules, the copy module, and the loader general file. In a subsystem, the AA portion of the VSN will be identical for all the DASD described in the preceding sentence. Across subsystems, the AA portion is required to be unique. It follows directly from the format that up to 10000 differing VSNs could be coded for any 1 subsystem; however, the capacity of the module file status table limits the actual number of usable DASD to 3999.

Six SIP parameters are used to code TPF VSNs. The first 5 are on the ONLFIL macro; they are VSNCHAR and VOLNO<sub>x</sub> (where *x* = A, B, C, or D). The sixth parameter is VOLNLGF and it is found on the GENSIP macro. VSNCHAR specifies the AA portion of the VSN for not only the online modules, but also the copy module and the LGF as well. VOLNO<sub>x</sub> and VOLNLGF code the NNNN portion for the online modules and the loader general file respectively. Only the starting VSN is needed as SIP will sequentially assign ascending numerals based on the number of modules in each device type.

The ONLFIL macro also provides a parameter for specifying the VSNs of spare devices to be used for copying online modules in the event of a device failure.

SPARVSNx (where x = A, B, C, or D) specifies the first 4 characters of VSNs to be ignored by disk roll call during IPL. This allows the user to attach spare devices to be renamed and copied at a later time.

For additional information concerning VSNs and how they are coded, see “ONLFIL” on page 282.

*Table 9. Disk Online Module List Example*

Symbolic Module Numbers	Volume Serial Number	Device Type
26	TP0110	3390
27	TP0111	3390
34	TP2120	3380
35	TP2121	3380

While AANNNN is the standard format, TPF provides a migration tool to convert VSNs that employ the older format: AAANNN. The old AAANNN format had to be obsoleted when TPF initiated support for over 1000 DASD. The migration tool is activated by SIP ONLFIL macro parameter OLDVSN. OLDVSN accepts the AAA portion of the old format as input. The first 3 characters of the old format VSN must be identical for all online DASD of the same subsystem. Across subsystems, the first 2 characters must be unique.

When OLDVSN is coded, the VSNs generated in keypoint V will be in the new AANNNN format, but a migration driver switch is turned on. The switch alerts the restart programs to convert the VSNs. When the old format is converted to the new format, the third character is changed to zero (for example, AAANNN → AA0NNN). In order for the migration tool to work, it is assumed that the table of online VSNs in the old keypoint V are in a constantly ascending sequence with no gaps in a device type. For additional information about how to code OLDVSN in order to activate the migration driver, see “ONLFIL” on page 282.

Table 10 is provided for recording all online disks. The starting volume serial number for each device type will be listed in the SIP report.

*Table 10. Volume Serial Number Worksheet*

Symbolic Module Number	Volume Serial Number	Device Type
02	TP0101	3380
<b>Note:</b> * For example, First 3380 ONLFIL VOLNOA=101, VSNCHAR=TP		

---

## Alternate Paths to Disk Storage

Sometimes when a program requests storage I/O, the pathway between the processor complex and disk storage is found to be busy because of an earlier I/O request on the same storage director (SD). When there is only 1 path defined through a storage director between the processor complex and disk storage, the program I/O request must wait for the earlier request to complete. When there is more than one path from a processor complex to disk storage, a program requesting storage I/O does not need to wait for an earlier request to complete. An alternate pathway can be selected automatically.

Pathways between the processor complex and disk storage are defined by the I/O configuration data set (IOCDS) for each installed processor. IOCDSs are created by an IBM-supplied I/O configuration program (IOCP) and are modified whenever there are any alterations required in the installed I/O configuration. For example, adding a new DASD controller would involve an addition to the IOCDS as part of the installation. The IOCP required for creating and maintaining IOCDSs is not a part of the TPF-supplied product. There is a separate IOCP associated with each IBM processor. For detailed IOCP operating information about a particular processor see *ES/9000, ES/3090 Input/Output Configuration Program User's Guide and ESCON Channel-to-Channel Reference* for the processor. For information about devices available on the System/370 processor, see *IBM System/370 Input/Output Configurator*.

The following are considerations that relate specifically to the use of alternate pathways to disk storage in TPF systems:

- The symbolic device address (SDA), which uniquely identifies an I/O device, will be either 370-like with 3 hexadecimal digits for communications, tape devices, and so on, or may be 4 hexadecimal digits (up to X'7FFF') found in 370/XA for MPIF and DASD devices. In order to simplify the construction of symbolic device addresses, the channel path identifier (CHPID), control unit (CU), and device number should be used to form the SDA. For example, an SDA for device 0 on CU C accessible from CHPID 4, would be 4C0.
- For single processor, non-loosely coupled systems with no DASD RPQs, each DASD has 1 subchannel defined. Multiple paths to a DASD are reflected in the subchannel information block (SCHIB) associated with that DASD. The IOCDS for this configuration is similar to that required by MVS and VM. In this configuration, the channel redrive RPQ is optional.
- When multiple paths are associated with a DASD, the lowest numbered CHPID should be used to form the SDA.
- For loosely coupled systems that use the limited lock facility (LLF), each DASD should be assigned a single path subchannel for each storage director (SD). For example, if a string of 16 devices is attached to a control unit (CU) with 2 SDs, each SD should have 16 subchannels defined for it. Each subchannel has a single path defined.

All DASD control units in a loosely coupled configuration require either of the following:

1. The limited lock facility (LLF) **and** static switch RPQs
2. The concurrency filter lock facility (CFLF).

Additionally, the channel redrive RPQ is required on the processors. CFLF can perform error recovery on another channel; therefore, the static switch RPQ is not used. However, CFLF still uses the channel redrive to prevent the CPU from becoming I/O bound.

---

## Permanent Record Area

Some system records that are essential for system restart are placed in a special area on the online disk modules, called the permanent record area. The permanent record area is unique because the data does not adhere to the structure of a general file, a general file data set, a fixed file, or a pool file.

The permanent record area includes the system programs necessary to start the IPL process. The IPL bootstrap program itself and the volume serial number (VSN) are placed on file storage by a standard MVS utility program (ICKDSF).

Closely associated with the permanent record area is the IPL area, core image restart area and the system keypoints, which are no longer physically located in the permanent record area itself. The ability to IPL the system relies on the ability to IPL a module that contains the permanent record area plus the IPL area, core image restart area, and system keypoints. The system operator may restart (IPL) the system by using any one of the modules of the first DASD device type in addition to IPLing from either the IPL prime or duplicate modules. The modules of the first DASD device type must be the same device type as the IPL prime and duplicate modules; The number of modules that can be IPLed are defined by the IPLABLE parameter on the ONLFIL macro.

The permanent record area is also made up of the 2 bootstrap records in addition to the volume serial number record. The permanent record area is loaded on the loader general file and the online modules by means of an MVS utility program (ICKDSF). The online modules are also formatted by means of the TPF offline formatter program (FMTR). These MVS initialization procedures are done before the initial IPL function.

The loader general file module is initialized and formatted with JCL provided by the SIP Stage II job stream. The online modules must be initialized and formatted by the user as a separate MVS procedure. The JCL necessary to do this is also provided in the SIP Stage II output. The format cards for the online modules are output by the FACE table generator. Support for the loader general file is unique from the standard general file support. The loader general file is not defined with a SIP GENFIL macro as are all other general file data sets. The SIP GENSIP macro parameters, VOLNLGF and LGFDV, are used to specify the loader general file volume serial number and device type, respectively.

Keypoint records are initially loaded during the IPL sequence by the general file loader online segment (ACPL) on the first prime module (this is the first module in the file status table) and also its duplicate module, if a duplicate module is indicated in the file status table.

The addresses of the areas in the permanent record area are generally imbedded in the keypoints and in the programs of the restart sequence. Table 11 provides a list of these areas and their permanent file addresses.



Table 11. Permanent Record Area Layout of the Restart Area

Cylinder	Track	Record	Record Type
0	0	1	In the Loader General File a standard 24-byte MVS bootstrap record is written using the device support facility (ICKDSF) utility program. In all the online modules the TPF bootstrap record is written using the TPF General File Loader (ACPL). In both cases the bootstrap record is known as IPL1.
0	0	2	The second MVS bootstrap record (IPL2) is also loaded by the MVS ICKDSF utility program on the Loader General File, and later replaced on all online modules by the TPF IPL2 record.
0	0	3	The volume label is initialized by the ICKDSF utility program on all modules.
0	0	4	The IPL program (IPLA) is written by the MVS ICKDSF utility program on the Loader General File only. The record does not exist in the online modules. The Image Pointer Record (CTKX) references IPLA in the online modules.
0	1	1	1KB of space is reserved for migration purposes for the former CTKX pointer.
0	1	2	The image control record (ICR) is written and initialized by ALDR with the first image entry being defined as primary.

The remainder of the core image area, including the image pointer record (CTKX) and the keypoints, are loaded by the general file loader online segment (ACPL).

Records in the core image area are not spread evenly across DASD modules like data records and other programs. Rather, the core image is loaded to a defined area on each module until the defined area is full and then overflows to the same area on the next disk. Note that #CTKX, #CIMRx, #IPLx, and #KEYPT are vertical records.

The core image restart area includes the following components:

- Control program (CP)
- Program Allocation Table (IPAT)
- In-core dump formatter (ICDF) program
- Online segment of the general file loader (ACPL)
- Global synchronization table (SIGT)
- Record identifier attribute table (RIAT)
- File address compute table (FCTB)
- USR1 (BSS only)
- USR2 (subsystem shared).

**Note:** The Image Control Record contains the loader general file and the online addresses of CTKX. IPLA and IPLB are part of IPL record #IPLx.

The record size for the CIMR components is 4K. Therefore, knowing the size of the control program, Program Allocation Table (IPAT), FACE Table (FCTB), Record ID Attribute Table (RIAT), System Interprocessor Global Table (SIGT), In-Core Dump

Formatter (ICDF), Online General File Loader (ACPL), and so forth, you can determine the number of records required. SIP calculates the relative record number of each core image area component and initializes the image pointer record (CTKX) with these values.

Table 12 shows how to calculate the number of #CIMR records. Users need to determine the number of records to code for #IPLx, #CIMR, and so on. This information is provided in Table 18 on page 111. The sizes noted were obtained from the TPF development test system and are only provided for the purpose of preliminary estimates. It is not necessary for the purpose of system design to know the exact sizes, but a reasonable estimate is required.

*Table 12. Online Restart Area Work Sheet (BSS or BSS in an MDBF System)*

Core Image Program	Approximate Size (Bytes)	Notes
CPS0	1 048 576	2
IPAT	520 000	5
ICDF	83 172	
ACPL	20 000	
SIGT	1 860	3
RIAT	12 080	4
FCTB	79 464	1
USR1	2	IBM default. The actual size depends on user-defined inputs.
USR2	2	IBM default. The actual size depends on user-defined inputs.
<b>Subtotal</b>	<b>1 613 772</b>	
<b>Note:</b> <b>Expansion Bytes (30%):</b> 484 132 <b>Total Number of Bytes Required:</b> 2 097 904 <b>Minimum Records (#CIMR1) across all Modules:</b> 515 <b>(Total ÷ 4079)</b>		

If you have a 10 prime module system (with 10 dupes) and you want 1 prime and 1 backup, copy the #CIMR1 code with 52 records. ( $52 \times 10 \geq 515$ )

**Notes:**

1. The total size of the FACE table depends on the number of RAMFIL macros coded by the user, with each record type requiring approximately 60 bytes of storage.
2. The resident control program is made up of both the mandatory and the optional CSECTs. The size of this area can vary depending on what functions are included in the system.
3. The SIGT table size varies based on user requirements to synchronize the global area.
4. The RIAT size depends on the number of RIATA macros coded in the RIAT source table.
5. The IPAT size depends on the number of programs, transfer vectors, and spares defined to the system. Each one takes about 96 bytes.

IPLA requires 4 #IPLx ordinals. IPLB can be assembled and its size can be obtained from the assembly. This size should be increased by 30% and the new size should be divided by 4079 to calculate the number of ordinals to allocate to #IPLx.

#CIMRx ordinals can be calculated by running a full SIP and inspecting SKGTSZ. This SIP-created segment contains the sizes of all the CIMR components. Add 30% to the total size in bytes and then divide by 4079 to determine the number of ordinals.

---

## Database Allocation

There are four considerations when allocating DASD space for the database.

1. What records are needed?
2. How much space will they occupy?
3. Which records need backup copies?

Backup copies are often referred to as *duplicates*, or more simply, *dupes*. All records, some records, or no records may have duplicates. Whether records are duplicated is determined by the overall DASD duplication scheme (whether the system is fully, selectively, or nonduplicated).

4. Where will the records be located?

Each record group must have a specific and unique address assigned to it.

The allocation of records on all online modules in the same logical device type (DEVA, B, C, and D) is identical.

See “Keypoint Records” on page 73 for the allocation of the keypoint area.

## Record Types and Space Requirements

In the disk file programs and data records area, the records are grouped as follows:

- File-Resident Application Programs
- Fixed File Data Records (including the 4K keypoint records)
- Pool Storage Area.

Fixed file data records in the system are mainly disk resident. The fixed file area may be formatted for large (1055-byte), small (381-byte), or 4K-byte records, which are either duplicated or nonduplicated. Records are allocated across the online modules. The number and size of each type of record is determined based on various system and application parameters.

Records allocated to other file areas are omitted from the fixed file data record area. In addition to the fixed file disk resident records, main storage (#GLOBA, and others) records must be allocated. Backup copies for these records are disk resident in the fixed file area. Space should also be included for at least a 20 % expansion factor in each fixed area. These expansion areas are defined by coding SPARE records in RAMFIL statements (see “RAMFIL” on page 290). Table 13 on page 92 is a worksheet for allocating fixed file data records. In order to convert the number of records to the number of tracks per module, divide the number of records of a given size by the number of records per track, times the number of online modules.

Table 13. Fixed File Data Record Worksheet

Fixed File Record Types	Number of Records	Number of Tracks/Modules
4KB Keypoint Records		
4KB, NonDup, High Activity		
4KB, NonDup, Low Activity		
Large, NonDup, High Activity		
Large, NonDup, Low Activity		
Small, NonDup, High Activity		
Small, NonDup, Low Activity		
4KB, Dup, High Activity		
4KB, Dup, Low Activity		
Large, Dup, High Activity		
Large, Dup, Low Activity		
Small, Dup, High Activity		
Small, Dup, Low Activity		

Disk file pool storage areas are used by system and application programs for the storage of temporary data. The directories which control the use of the pool storage areas reside in the fixed file area. Directories exist for each of the following pool types.

- 4K long-term
- Large long-term
- Small long-term
- 4K long-term duplicated
- 4K long-term duplicated FARF6
- Large long-term duplicated
- Small long-term duplicated
- 4K short-term
- Large short-term
- Small short-term.

Whether short-term pools are duplicated or not depends on the system's file configuration. DASD short-term pools are only duplicated in a fully duplicated file system.

After fixed file requirements have been determined, the balance of the file space is available for the assignment of random pools. Some of the things that should be considered in the assignment of the file area for file pool storage are:

- Requirements for a given pool can only be estimated at system initialization time. It is suggested that you do not allocate all available pool space at system initialization but hold some area in reserve for later addition to a given pool as dictated by actual needs.
- A particular pool type does not have to consist of only one contiguous interval on disks. Pool segments are supported in which different areas of disks are tied together as 1 pool through the directory scheme.

## UFT/FTI Examples

The heart of the expanded addressing scheme represented by FARF4, FARF5, and FARF6 is the UFT/FTI. The UFT/FTI design uses a hierarchical technique for dividing very large amounts of file address space. A UFT points to several FTIs and each FTI potentially points to a split in the FACE table. The UFT and FTI are part of a file address. Together with a record ordinal number, a UFT/FTI combination specifies a file address in the database.

The UFTI parameter on the UFTFTI statement specifies a UFT followed by the FTI size. The FTI size specifies a length, in bits, of FTIs to be coded for a particular UFT. It is not the FTI itself. The UFT/FTI pairs coded on the UFTI parameter are used only for FARF4 and FARF5 addresses. The UFT/FTI pairs coded on the UFTI6 parameter are used only for FARF6 addresses.

For example:

```
UFTFTI  UFTI=((3,8),(4,12))
```

specifies a slice starting at UFT 3 comprising up to 256 (that is,  $2^8$ ) FTIs, and a slice starting at UFT 4 comprising 4096 FTIs.

For example:

```
UFTFTI  STAGE=FARF45,MODE=FARF5,
        UFTI=((0,8),(1,12),(2,10),(61,16),(62,14),(63,12),
        (20,23),(21,25)),
        UFTI6=((2,10),(30,15))
```

**Note:** FARF6 addresses are dispensed in either FARF34 or FARF45 address mode.

The meaning of the UFTI parameter on this statement is:

- UFT 0 has an 8-bit FTI associated with it. This means there are  $2^8$  (256) UFT/FTI address pairs that may be used.
- UFT 1 has a 12-bit FTI associated with it. This UFT/FTI combination specifies 4096 UFT/FTI pairs.
- UFT 21 has a 25-bit FTI associated with it. This combination provides 33 554 432 UFT/FTI pairs.
- The UFTI6 parameter defines two UFT/FTI pairs 2,10 and 3,15. The same UFT can be used for FARF4/5 addresses as well as for FARF6 addresses.

The FTI size, in bits, defines the number of valid FTI numbers that can be associated with a particular UFT. In turn, the UFT/FTI combination limits the number of ordinals that can be represented by the combination. Because the number of UFT bits is fixed (at 6 bits for FARF4 and FARF5 and 16 bits for FARF6), once the FTI length is set, the maximum number of ordinals for the record type is also set. The maximum number of ordinals is represented by the remaining bits in FARF5 or FARF6 addresses and by the remaining bits minus 2 additional control bits in FARF4 addresses. While FARF addressing is described in detail in *TPF Database Reference*, FARF4, FARF5, and FARF6 addresses are distinguished graphically by Figure 23 on page 94.

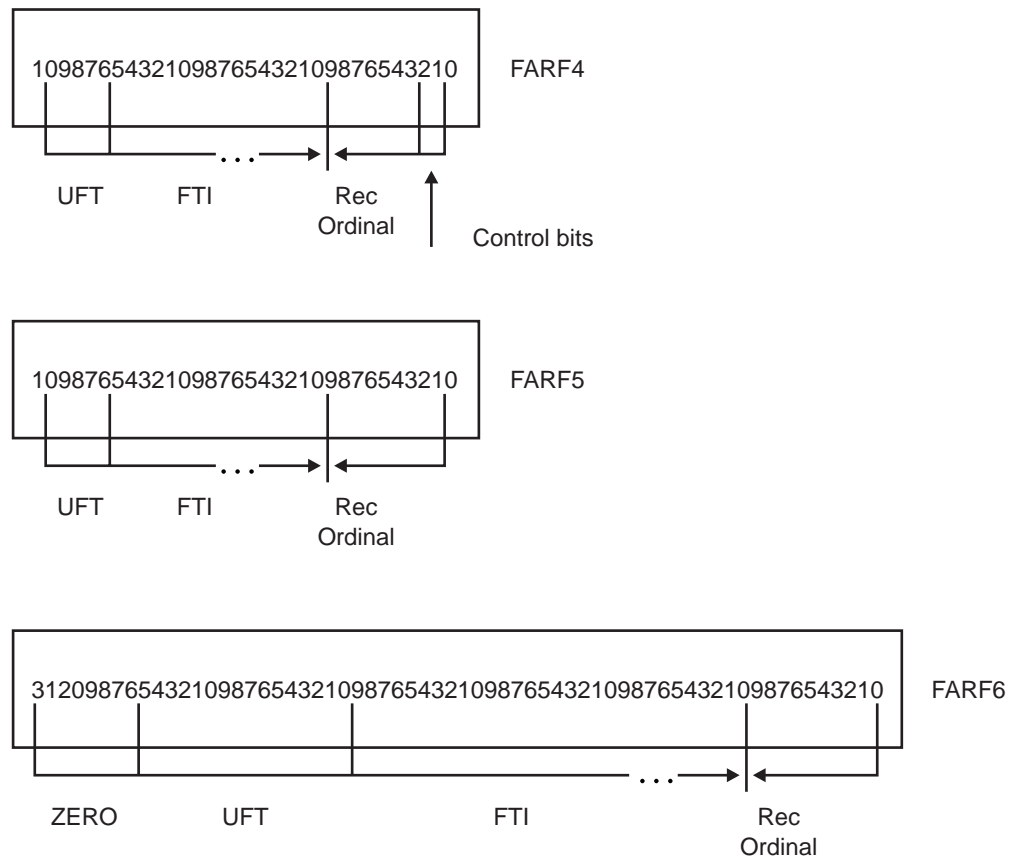


Figure 23. FARF4, FARF5, and FARF6 Addresses

Ellipses (...) in the figure represent the variable nature of the FTI.

UFTI pairs can be defined for both FARF4 and FARF5. The UFTI4 and UFTI5 parameters on the RAMFIL statement contain actual UFT,FTI pairs. The following is valid assuming the previous UFTFTI statement:

```
RAMFIL TYPE=LSA,RECNO=00014,RECID=#APPOP,DUPE=YES,BAND=1616,
      BASE=00800,UFTI4=(61,20),UFTI5=(61,21)
```

The UFTI4 parameter states that the FARF4 address for record ID #APPOP has 61 as a UFT and 20 as an FTI. UFT 61 is defined earlier as having a 16 bit FTI, so all FTIs ranging between 0 and  $2^{16} - 1$  (65 535) are valid. The UFTI5 parameter defines a FARF5 address for #APPOP with a UFT of 61 and an FTI of 21. Note that the same UFT can be used for both FARF4 and FARF5 addresses but a UFT/FTI pair can only be used once.

Here is another example:

```
UFTFTI UFTI=((10,6),(11,6))
RAMFIL RECID=#FRED,RECNO=100,BAND=10,UFTI4=(10,1),UFTI5=(11,1)
RAMFIL RECID=#BILL,RECNO=101,BAND=11,UFTI4=(10,2),UFTI5=(11,2)
RAMFIL RECID=#EDNA,RECNO=102,BAND=12,UFTI4=(10,3),UFTI5=(11,3)
```

UFT 10 has an FTI length of 6 bits, so there are  $2^6$  (64) FTIs pointed to by UFT 10. The maximum number of ordinals in each slot is as follows:

```
32 bits
- (6 bits UFT length + 6 bits FTI length)
-----
20 bit ordinal number or ordinals <  $2^{20}$  (1 048
```

or  
 - (2 FARF4 control bits)  
 -----  
 18 bit ordinal number or ordinals < 2<sup>18</sup> (262 144) for F

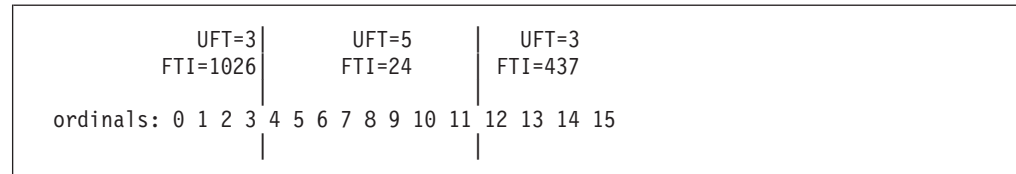
UFTs and FTIs can be thought of as first and middle names, while ordinals can be considered last names. Every unique record ordinal needs a unique first, middle, and last name.

For example:

```
UFTFTI  UFTI=((3,22),(5,21)),MODE=FARF4
RAMFIL  RECID=#FRED,RECNO=16,UFTI4=((3,1026),(5,24),(3,437)),TYPE=S SA
```

A series of record ordinals is specified by the following RAMFIL statement. The type of record is #FRED. There are 16 small ordinals on device A. UFTI4 indicates they are FARF4 record ordinals that are divided among 3 UFTI pairs. The first pair (3,1026) is the first and middle name for the first group of four ordinals. The last names for these are 0, 1, 2, and 3. The second group of ordinals is specified by the (5,21) pair. The second group has 3 bits available according to the UFTFTI macro above and this allows 8 ordinals: 0, 1, 2,..., and 7. Finally, the third group has 2 bits, because it is governed by the UFT 3, and so 4 ordinals.

Shown graphically, the 16 ordinals specified by this RAMFIL statement are distributed over 3 UFTI pairs.

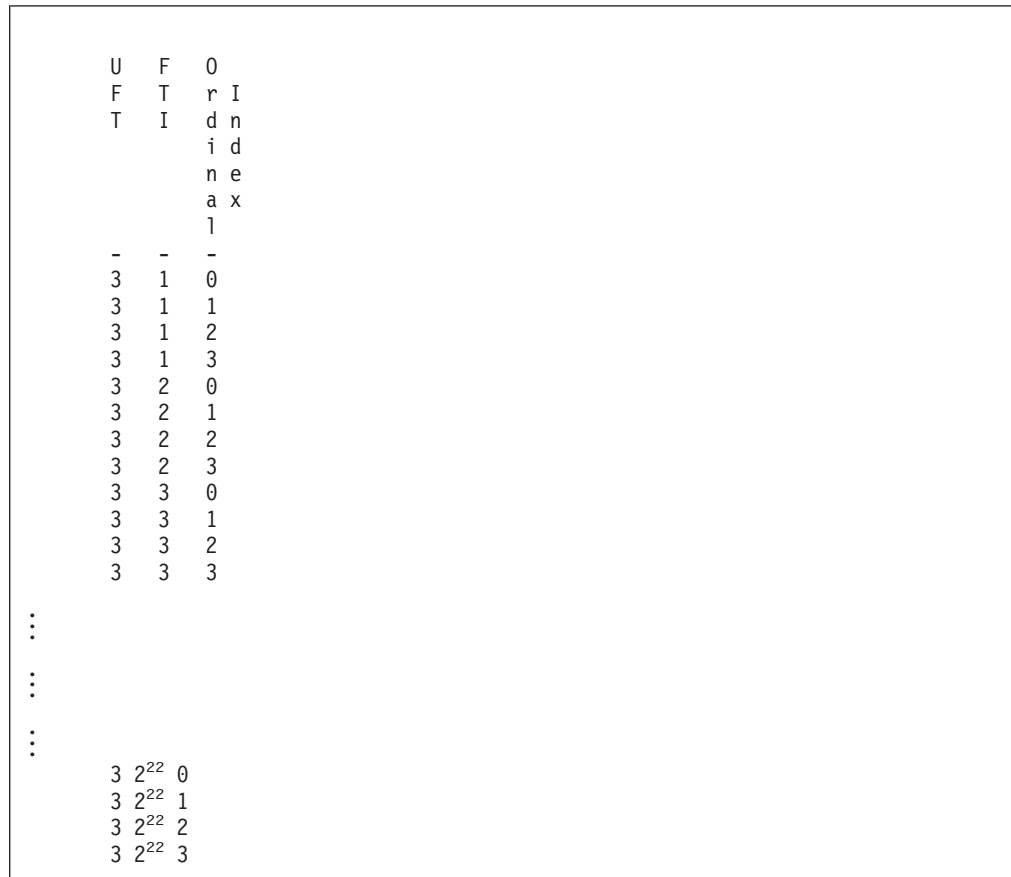


Written out, the addresses for the ordinals are:

#FRED	ordinal	0:	3	1026	0	
	ordinal	1:	3	1026	1	
	ordinal	2:	3	1026	2	
	ordinal	3:	3	1026	3	_____
	ordinal	4:	5	24	0	
	ordinal	5:	5	24	1	
	ordinal	6:	5	24	2	
	:					
	:					
	:					
	:					
	:					
	:					
	ordinal	11:	5	24	7	_____
	ordinal	12:	3	437	0	
	ordinal	13:	3	437	1	
	ordinal	14:	3	437	2	
	ordinal	15:	3	437	3	

Observe that when a new UFTI pair starts, the ordinal number index in the address begins again at zero.

Thinking about UFTI pairs and ordinal numbers as names helps to clarify how UFTFTIs are structured. Each first and middle name pair specified by the UFTI pairs implies a limited set of ordinal indexes, as shown earlier. Moreover, each first name has a large set of middle names that go with it. As with (3,22), the previous example we have  $2^{22}$  possible middle names. Putting these middle names together with the ordinal index results in a diagram of addresses as follows:



You could make a similar diagram for the pair (5,21), with the only exception being the last group of ordinal indexes distinguished by their middle names is as follows:



```

U   F   0
F   T   r I
T   I   d n
           i d
           n e
           a x
           l
-   -   -
:   :   :
5 221 0
5 221 1
5 221 2
5 221 3
5 221 4
5 221 5
5 221 6
5 221 7

```

## Two Unusual Cases

Choosing appropriate FTI sizes for each UFT during coding of the UFTFTI statement is important. When a large FTI size is chosen, only small ranges of ordinal numbers can be defined for each FTI. Suppose you code the following:

```
UFTFTI  STAGE=FARF45,MODE=FARF5,UFTI=(2,25)
```

The UFT/FTI pairs defined this way only map 2 ordinal numbers each. This is because of the following arithmetic based on the structure of UFT/FTI addresses:

6 bit UFT (fixed amount)	32 bit address
+ 25 bit FTI (from UFTI parameter)	- 31 bits (UFT + FTI)
---	---
31 bits accounted for (UFT + FTI)	1 bit ordinal numbers

The 2 ordinal numbers are 0 and 1.

Because #FRED is 10 ordinal numbers long, 5 UFT/FTI combinations are required, each containing 2 records. This means that you would have to code:

```
RAMFIL  RECID=#FRED,RECNO=10,...,UFTI5=((2,0),(2,1),(2,2),(2,3),
(2,4))
```

The UFTI5 definition contains specifications for all the UFT/FTI pairs needed for the 10 records.

Such UFT/FTI combinations are not bad, but you can imagine writing large numbers of combinations for large groups of records using inappropriate UFT/FTI combinations. Mapping a pool type containing a large number of ordinals with a 1-bit area for representing records is extremely difficult, time-consuming and error prone. Moreover, because each FTI takes up about 8 bytes of storage in the online FCTB, resulting in a needlessly large FACE table.

The opposite extreme is:

```
UFTFTI  STAGE=FARF45,MODE=FARF5,UFTI=(3,1)
```

and the following RAMFIL:

```
RAMFIL  RECID=#FRED,RECNO=10,...,UFTI5=(3,0)
```

This UFTFTI macro provides for 2 UFT/FTI pairs. This means that each UFT/FTI combination maps 25 bits of ordinal, or 33 554 432, records. Obviously, only one UFT/FTI combination is needed to map all 10 ordinals of #FRED. This results in a waste of  $2^{25} - 10$  addresses (33 554 422), because the UFT/FTI combination (3,0) can only be used for a single RAMFIL statement.

A small FTI size is good for mapping pool or fixed record types containing a large number of ordinals. Large FTI sizes are good for mapping pool or fixed record types with a small number of ordinals. By being aware of the record type requirements it is possible to waste very little addressing capacity and yet have a minimal online FACE table size.

If a particular record type requires many ordinals, there must be a sufficiently small FTI to provide the bits required to represent the ordinals. If a record type will only have a few ordinals, the FTI size can be very large and there will still be enough bits to represent the highest ordinal in the record type. There are no performance considerations for laying out UFT/FTI combinations, so the UFT/FTI combinations and record ordinal requirements can be balanced in this way without performance considerations.

## Record Duplication

There are two kinds of duplication:

- Device duplication, specified by the DUPTYPx parameter on the ONLFIL macro
- Record duplication, specified by the DUPE parameter on RAMFIL statements.

These combine into three kinds of duplicated systems:

- Nonduplicated (no records have duplicates)
- Fully duplicated (all records have duplicates)
- Partially (or selectively) duplicated (some records have duplicates).

The duplicate record in a record type (either pool or fixed) is assigned to the same relative position on an alternate disk module. Therefore, an even number of modules is required.

Device duplication is sometimes referred to as system duplication because customers often run with only 1 device type (DEVA). If more than 1 device type is defined, the duplication on 1 device type can be different from the duplication on another device type. On a fully duplicated device, all records are duplicated no matter how the DUPE parameter is coded on any RAMFIL. This is true for both fixed and pool records.

The partially duplicated database, like a fully duplicated system, requires an even number of modules. However, within a partially duplicated database, nonduplicated long-term pools are supported. In a partially duplicated database:

- Short-term pools and long-term, non-duplicated pools are spread across all device modules (prime and duplicate).
- For nonduplicated fixed records, the records are spread across only the prime modules. The corresponding area on the duplicate module is unused.

See Figure 24 on page 102 for an example. The DUPE parameter on the RAMFIL statement controls whether the record type is duplicated or not.

A nonduplicated database implies a system with no duplicated record types. Records on a nonduplicated device type are never duplicated, no matter how the DUPE parameter is coded on any RAMFIL. This is true for both fixed and pool records.

For fully duplicated and nonduplicated device types, the DUPE parameter is meaningless, except for pools. The DUPE parameter controls the pool type being defined out of the 9 possible pool types. A 4K, long-term, duplicated pool is a different pool from a 4K, long-term, nonduplicated pool type even though both are duplicated in a fully duplicated system.

## Address Assignment

After the method of duplication has been selected, the next step is to assign specific disk addresses to the various record types. It is recommended that the file design adhere to the following guidelines:

- Highly accessed records should be located under the read/write heads as near to the center point of arm excursion as possible.
- The balance of record types should be allocated on either side of this center point in descending order of activity.
- Expansion areas should be provided to allow for growth and oversights.
- Programs and most fixed file data records should be duplicated.
- Fixed file and pool file areas may be segmented. Multiple RAMFIL statements may be coded for the record type.

All disk packs of a logical device type (DEVA, DEVB, and so forth) are formatted the same. Therefore, only the disk layout for 1 module needs to be determined.

## File Design Worksheets

Table 14 provides a disk file design worksheet for a partially duplicated file layout (module-to-module duplication). The worksheet represents a single prime module. Duplicate modules require that all duplicate copies of prime records are to be located in the same relative position because they are on the prime modules. All fixed records that are specified as nonduplicated (fixed nonduplicated or fallback) on the prime modules would also be specified as nonduplicated on the duplicate packs but would contain no data. Nonduplicated pools should be spread across all modules, therefore utilizing the space available on the duplicate modules.

Enter the total number of records for each record type and calculate the size of each area (number of tracks per module). The size of each area equals the number of full tracks required for a given area divided by the number of prime modules. Users should be sure to provide spare tracks for future expansion. At least 10 % per record type, where appropriate, is recommended.

*Table 14. Module-to-Module Duplication Worksheet*

Record Type	Duplicate File?		Address HEX	
			CYL	HD
Permanent Records <sup>1</sup>	00	00	00	00
Core Image Restart Area <sup>1</sup>	00	xx <sup>2</sup>	00	xx <sup>2</sup>

1. For a description see "Permanent Record Area" on page 88.

2. These records may be located anywhere.

Table 14. Module-to-Module Duplication Worksheet (continued)

Record Type	Duplicate File?		Address HEX	
			CYL	HD
4KB Keypoints <sup>1</sup>				
Small Long-Term Pool				
Large Long-Term Pool				
4KB Long-Term Pool				
Large Short-Term Pool				
Small Fixed				
Small Programs				
Large Programs				
4KB Programs				
Large Fixed				
4KB Fixed				
Small Short-Term Pool				
Large Fixed				
Small Long-Term Pool				
Large Long-Term Pool				

## Partially Duplicated Layout Example

Once the disk files are completely defined and the duplication factor is determined, determine the starting address of each area and complete the worksheets.

Figure 25 on page 103 provides example SIP RAMFIL statements for the sample file layout in Table 15. Note that the sample is for a 4-module, selectively duplicated system, and the keys relate to locations in Figure 24 on page 102.

Table 15. Four-Module 3380 Sample File Layout

Record Description	Dupe ?	K E Y	Total # Records	Tracks per Module	DISPL	Decimal		Hex	
						CYL	HD	CYL	HD
Permanent Records	<sup>3</sup>	A		1	0	00	00	00	00
Small LT Pool	No	B	817,260	3,855	25	01	10	01	0A
Large LT Pool	No	C	114,600	955	3,880	258	10	102	0A
4KB LT Pool	No	D	16,080	402	4,835	322	05	142	05
Small Fixed	No	E	3,604	34	5,237	349	02	15D	02
Small LT Pool	Yes	F	21,200	200	5,271	351	06	15F	06
Large LT Pool	Yes	G	16,080	268	5,471	364	11	16C	0B
4KB LT Pool	Yes	H	9,000	450	5,739	382	09	17E	09
Small Fixed	Yes	I	27,348	258	6,189	412	09	19C	09
Large Fixed	Yes	J	15,480	258	6,447	429	12	1AD	0C
4KB Fixed	Yes	K	5,160	258	6,705	447	00	1BF	00

3. For a description, see "Permanent Record Area" on page 88.

Table 15. Four-Module 3380 Sample File Layout (continued)

Record Description	Dupe ?	K E Y	Total # Records	Tracks per Module	DISPL	Decimal		Hex	
						CYL	HD	CYL	HD
Small Program	Yes	L	742	7	6,963	464	03	1D0	03
Large Program	Yes	M	2,040	34	6,970	464	10	1D0	0A
4KB Program	Yes	N	3,000	150	7,004	466	14	1D2	0E
Large Fixed	No	O	102,000	1,700	7,154	476	14	1DC	0E
Small ST Pool	No	P	21,200	100	8,854	590	04	24E	04
Large ST Pool	No	Q	16,080	134	8,954	596	14	254	0E
4KB ST Pool	No	R	9,000	225	9,088	605	13	25D	0D
4KB Keypoints	<sup>3</sup>	S	240	12	9,313	620	13	26C	0D

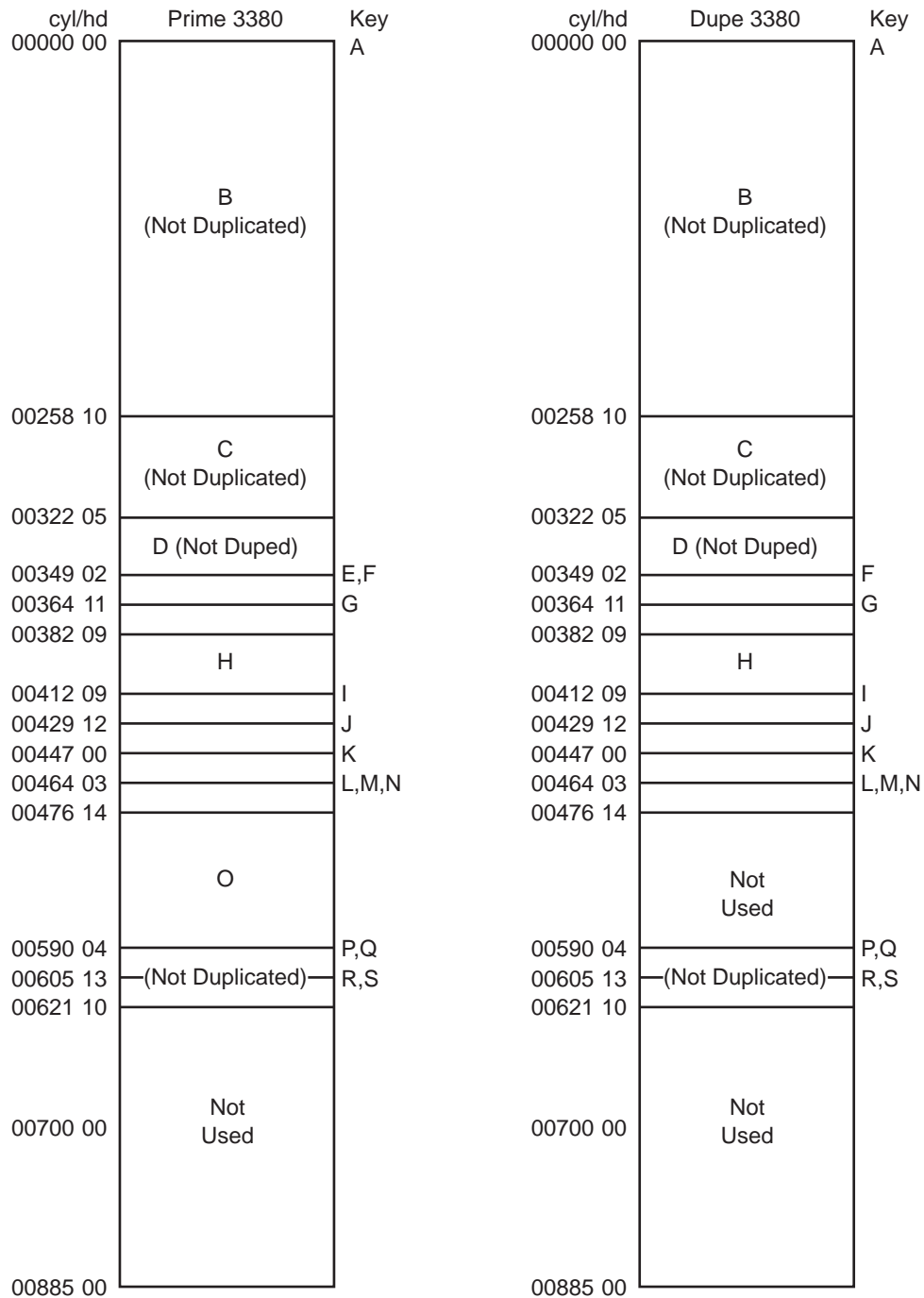


Figure 24. 3380 Device Sample Layout. This diagram is based on the information contained in Table 15 on page 100.

```

RAMFIL  RECID=POOL,TYPE=SSA,RECNO=817260,DUPE=NO,POLID=LT,BASE=00110
RAMFIL  RECID=POOL,TYPE=LSA,RECNO=114600,DUPE=NO,POLID=LT,BASE=25810
RAMFIL  RECID=POOL,TYPE=4SA,RECNO=16080,DUPE=NO,POLID=LT,BASE=32205
RAMFIL  RECID=#RID01,TYPE=SSA,RECNO=1802,DUPE=NO,BASE=34902
RAMFIL  RECID=#RID02,TYPE=SSA,RECNO=1802,DUPE=NO
RAMFIL  RECID=POOL,TYPE=SSA,RECNO=21200,DUPE=YES,POLID=LT,BASE=35106
RAMFIL  RECID=POOL,TYPE=LSA,RECNO=16080,DUPE=YES,POLID=LT,BASE=36411
RAMFIL  RECID=POOL,TYPE=4SA,RECNO=9000,DUPE=YES,POLID=LT,BASE=38209
RAMFIL  RECID=#RID03,TYPE=SSA,RECNO=20000,DUPE=YES,BASE=41209
RAMFIL  RECID=#RID04,TYPE=SSA,RECNO=7348,DUPE=YES
RAMFIL  RECID=#RID05,TYPE=LSA,RECNO=15480,DUPE=YES,BASE=42912
RAMFIL  RECID=#RID06,TYPE=4SA,RECNO=5160,DUPE=YES,BASE=44700
RAMFIL  RECID=#XYZZZ,TYPE=SSA,RECNO=742,DUPE=YES,BASE=46403
RAMFIL  RECID=#XYNLL,TYPE=LSA,RECNO=2040,DUPE=YES,BASE=46410
RAMFIL  RECID=#MRES4,TYPE=4SA,RECNO=3000,DUPE=YES,BASE=46614
RAMFIL  RECID=#RID07,TYPE=LSA,RECNO=102000,DUPE=NO,BASE=47614
RAMFIL  RECID=POOL,TYPE=SSA,RECNO=21200,DUPE=NO,POLID=ST,BASE=59004
RAMFIL  RECID=POOL,TYPE=LSA,RECNO=16080,DUPE=NO,POLID=ST,BASE=59614
RAMFIL  RECID=POOL,TYPE=4SA,RECNO=9000,DUPE=NO,POLID=ST,BASE=60513
RAMFIL  RECID=#KEYPT,TYPE=4SA,RECNO=240,DUPE=NO,BASE=62013

```

Figure 25. Sample Device RAMFIL Macros. Not all parameters are shown (see “RAMFIL” on page 290).

Database configurations containing multiple DASD device types require separate file layouts and separate RAMFIL macros for each device type. The appropriate worksheets provided in this section should be used for each device type included in the system being generated.

## System Utilized Record Types

The tables in this section contain lists of the fixed file data record types, including main storage resident records (which have their backup copies on file) that are required by the system programs. In addition to these system records, users will have their own application fixed file data records to include in their database. It should be noted that several of the different system record types are associated with various system packages and components, which are optional. For example, the various records associated with MPIF are only required if MPIF support is defined via the various SIP macros.

Each fixed file record type, including both system and application record types, requires a SIP RAMFIL macro. Inputs to the RAMFIL macro include the FACE record type and the number of records to be allocated. A minimum of 10 % extra records (coded as SPARES in RAMFIL macros) is recommended to allow for expansion. Lists of system utilized data records are provided in Table 16 and Table 18.

Use care when adding or deleting SPARE records. The IPAT and #PROGn area must be synchronized. IBMPAL is the segment used to allocate ECB-controlled programs. It creates both the IPAT and the SAL table. It also assigns program ordinal numbers in #PROGn. Ordinal numbers are assigned in the order that programs are coded. Once a program is assigned an ordinal number, it is important that the assignment never change. If a program ordinal number is changed, that program must be reloaded and all programs that enter it without using the NAME= parameter must be reloaded. For example:

IBMPAL	#PROG1 ordinal
SPARE OLD1	0
COSY,FR,OPTIONS=(KEY0,MONTC,RESTRICT)	1
BXAX,FR,FUNC=SBFCAP	2

CCKP,FR,OPTIONS=(KEY0,MONTC,RESTRICT)	3
COHA,FR,OPTIONS=(KEY0,MONTC,RESTRICT)	4
BWMS,CR,OPTIONS=(KEY0,MONTC,RESTRICT)	5
BWMU,TV,BWMS,1	6

In this example, if CCKP is obsoleted, it should **not** be deleted. CCKP should be replaced with a SPARE. If it is not replaced with a spare, COHA becomes ordinal 3 and every program allocated after COHA gets assigned a different #PROG1 ordinal number (PROG ordinal shift).

The same problem occurs if a new segment is added instead of replacing a SPARE. All segments below the added segment get a new ordinal number.

In most cases the proper way to add a segment is to find a SPARE and change it. The proper way to delete a segment is to replace it with a SPARE.

The analysis is more complicated when the program being deleted or added contains a FUNC= or an SS= switch.

- If a program allocated with SS=BSS is replaced by a SPARE, there will be no #PROG1 ordinal shift in a BSS (or base-only) system but there is a shift in any non-BSS subsystem. This is because the same IBMPAL deck is used to generate all the subsystems. When the BSS-only program was allocated, it had an ordinal number in the BSS but not in the other subsystems (where it did not exist). Consequently, the program allocated next after the BSS-only program has a different ordinal number in the BSS than it does in the subsystems.
- When SPARE is substituted for the BSS-only allocation, this is picked up by all the subsystems and the program allocated next after the SPARE has its ordinal number changed from what it was previously in the IPAT.
- However, the #PROG records contain programs based on the old IPAT. In order to correct this ordinal number mismatch the #PROG records need to be loaded with programs based on the new IPAT. This usually means a full load.

A similar kind of example could be described for the FUNC parameter that specifies function switches.

#PROGn is initialized when you do a full load. What program goes into a given #PROG ordinal is controlled by the IPAT. The IPAT is created from IBMPAL and any user input decks. To avoid full loads for this reason manage SPAREs carefully.

Many of the data records require initialization before starting up the system. The initialized data records are generated on 1 or more pilot tapes, which are loaded onto the online packs as discussed in “System Generation Process” on page 15.

The records requiring initialization fall into three basic categories:

1. Records needing complete initialization.
2. Records needing header and some variable information to be initialized.
3. Records needing only header information to be initialized. All core resident data records (for example, globals) must have at least a header initialized.

For categories 2 and 3, only 1 skeleton record must be initialized; for example, a standard data/message file (SDMF), and the required number can be generated on a pilot tape via the facilities of the system test compiler (STC).

The contents of the first 4 bytes of a data record (standard header) are:



**Bytes 1–2** The record ID, which was coded as any self-defining term, for example:

X'2100'

B'0011000011001100'

C'WW'

2134 (decimal number)

**Byte 3** The record code check

**Byte 4** The control byte

Table 16 contains a list of the system-required fixed file data record types arranged by system function. Included is the following information:

- Record macro names, which are the names of the macro DSECTs located in the macro release library.
- RIAT ID, which is contained in the first 2 bytes of the data record header, is defined in SIP through the RAMFIL ID parameter.
- Record size (1055, 381, or 4K bytes).
- FACE record type for use in the RAMFIL macro.
- For MDBF users of the HPO feature, an additional column has been added to define which subsystem the record should be allocated to. There are three possibilities:

**BSS** Allocate only to the basic subsystem.

**SS** Allocate only to the unique subsystem that requires the support (may be the BSS).

**BSS/SS** Allocate to the BSS and to all subsystems.

- A number of the records identified are only required if the support named under the Functional Area column is requested by the user through SIP.

Also indicated is the offline (MVS) program used to initialize each data record type. TPF provides the following offline programs for system record initialization:

- OSTG, for SNA communication tables.
- SCK GEN, for non-SNA communication tables.
- SIP, for keypoint records and message router tables.
- ASLOL, for 3270 mapping and paging records.
- STC, for most other records. STC is also used to initialize most application data records. Alternatives are user-written application programs (utilities) that are executed at restart time to initialize application records.

Table 16. System-Required Fixed File Record Types

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#ASMSI	AS4MF	D4C1	L	ASL	Mapping Paging	BSS/SS	
#BKDRI	BK0RP	C2D2	L	BRS0	Recoup Descriptor Records	BSS/SS	
#CANRI	MT0MT	D4E3	S	STC	UI Canned Messages	BSS/SS	
#CCBRU	ICCB	FF14	4KB	Online	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>

Table 16. System-Required Fixed File Record Types (continued)

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#CCEPS	CC0CC	C3C3	L	CCSSP	3705 Load /Dump	BSS/SS	
#CFREC	ICFST, ICFSB	FC2E	4KB	ZIFIL	Coupling Facility	BSS	<ul style="list-style-type: none"> <li>If you are not using coupling facility support, there is no need to allocate this fixed file record.</li> <li>A record code check (RCC) of 'X'01' is required for ICFST.</li> <li>A RCC of 'X'02' is required for ICFSB.</li> </ul>
#CF2LR	ICFLR, C\$CFLR	FC2F	4KB	ZCFLK INIT	Coupling Facility	BSS	If you are not loosely coupled, there is no need to allocate this fixed file record.
#CIMR1	IDSCDR	FC01 - FC0A	4KB	ALDR	Core Image Restart area for Image 1	BSS/SS	
#CMSIT	ISIDE	FF09	4KB	ZNSID	SNA Comms.	BSS/SS	See <i>TPF ACF/SNA Data Communications Reference</i>
#CN1ST	CN1ST	FC81	S	ZMIGR or Restart	Subsystem state table.	BSS	
#CO1RI	CO1DR	C3F1	L	STC	Unsolicited Message Processor (UMP)	BSS	
#CTKX	CX0CK	00ED	4KB	ALDR	Image Pointer Record	BSS/SS	
#CY2CPY	CY2KT	FF26	4KB	Pool Directory Capture	File Pools	BSS/SS	
#CY2KT	CY2KT	FF26	4KB	Pool Gen	File Pools	BSS/SS	
#CY2NEW	CY2KT	FF26	4KB	Pool Gen	File Pools	BSS/SS	
#DBRRI	DB0DB	C4C2	4KB	Online	Database Reorg.	BSS/SS	
#DSCRU	DSEQU	C4E2	L or 4KB	Online	General Data Set and General File Support	BSS/SS	
#FLOCK		FC2A	4KB	Online	File System	BSS/SS	
#GLOBL	GLOBx	C7D3	L	STC	Globals	BSS/SS	See <i>TPF System Installation Support Reference</i>
#HDREC	CB8HD	00E2	4KB	Online	MPIF	BSS	
#IBMML		Various	L		IBM Misc.	BSS/SS	See the code for the FACE table generator (FCTB) program FTVA03 for information about the number of ordinals to allocate for this fixed file record type.

Table 16. System-Required Fixed File Record Types (continued)

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#IBMMP4		Various	4KB		IBM Misc.	BSS/SS	See the code for the FACE table generator (FCTB) program FTVA03 for information about the number of ordinals to allocate for this fixed file record type.
#IBMMS		Various	S		IBM Misc.	BSS/SS	See the code for the FACE table generator (FCTB) program FTVA03 for information about the number of ordinals to allocate for this fixed file record type.
#IBMM4		Various	4KB		IBM Misc.	BSS/SS	See the code for the FACE table generator (FCTB) program FTVA03 for information about the number of ordinals to allocate for this fixed file record type.
#IDOTX	IDOTB		4KB		ZIDOT	BSS	
#IMQCK			4KB	Online	MQSeries Queue Manager	SS	MQSeries checkpoint records. The suggested number of #IMQCK records is at least twice the number of allocated SWBs.
#INODE	INODE	FC2A	4KB	Online	File System	BSS/SS	
#IPL1			4KB		IPLA/IPLB for Image 1	BSS	
#IPRTE	IPRTE	FC44	4KB	IP Restart	TCP/IP Native Stack Support	BSS	See <i>TPF ACF/SNA Network Generation</i> and the ZTRTE command in <i>TPF Operations</i>
#IZERO		FC2A	4KB	Online	File System	BSS/SS	
#KBA	IDSKPT	C3D2	4KB	ACPL	Keypoint backup area	BSS/SS	Keypoint backup area
#KEYPT	IDSKPT	D2D7	4KB	ACPL	Keypoint Record	BSS/SS	
#KSA1	IDSKPT	C3D2	4KB	ACPL	Keypoint staging area for image 1	BSS/SS	
#LKIBR	CM8CM	C3D4	S	03-CMR	SLC	BSS/SS	
#MAILxx		FC55	4KB		TPF Internet mail server support	SS	If you are using TPF Internet mail server support, see note 1 on page 110. Otherwise, there is no need to allocate this fixed file record.
#MQICD	C\$MQCD	00F5	4KB	Online	Message queue interface client support	BSS	See ZMQID DEFINE in <i>TPF Operations</i>
#NCBRI	NC0CB	C3C2	S	Online	SNA Comms.	BSS	

Table 16. System-Required Fixed File Record Types (continued)

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#OLDx			4KB		E-Type Loader Directory Records	BSS/SS	See <i>TPF System Installation Support Reference</i> . Required for E-type loaders.
#PDREC	CB9PD	00E1	4KB	Online	MPIF	BSS	
#PDREU	CB9PD	00E1	4KB	Online	MPIF	BSS	
#PROG1	IDSPRG	00FF	4KB	ACPL	E-type program area for Image 1	BSS/SS	
#PRORI	PR1OT	D6E3	L	STC	Processor Resource Owner Facility		See <i>TPF Main Supervisor Reference</i>
#PSTXCUR	CY7PL	FC39	4KB	Pool Gen	File Pools	BSS/SS	
#PSTXNEW	CY7PL	FC3A	4KB	Pool Gen	File Pools	BSS/SS	
#PTKST	CPTIC	D7D2	S	SCK GEN	Non-SNA Comms.	BSS/SS	
#PTSCK	SCKDS	E2D2	L	SCK GEN	Non-SNA Comms.	BSS/SS	
#PVR1	IDSPVR	FF0E	4KB	ACPL	PVR for Image 1	BSS/SS	PVR1 is required. RCC of X'FF' is required.
#RC8RFS	I80I8	FC33	L	ZPOOL INIT	File Pools or PDU	BSS/SS	
#RCATU	RC0AT	D9C3	L	SIP	Message Router	BSS	
#RCBRA	CI0CO	C3D6	L	STC	Non-SNA Comms.	BSS/SS	
#RLOG 1–32	ICRCT	FC27	4KB	Log recovery restart	Recovery log support	BSS, SS	For TPF transaction services, the recovery log fixed file records are always allocated to the BSS. Recovery log records may also be allocated to any user-defined application subsystem. The minimum allocation in the BSS and other subsystems is 60 tracks (see note 33).
#RRTRI	RR0RT	E2F2	4KB	OSTG	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#RV1RU	RV1VT	E2F6	4KB	OSTG	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#RV2RU	RV2VT	E2F7	4KB	OSTG	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#SATRU	SA0AT	E2F1	L	OSTG	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#SC1RU	ISCB	FF22	4KB	Online	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#SC2RU	ISCB	FF23	4KB	Online	SNA Comms.	BSS	See <i>TPF ACF/SNA Network Generation</i>
#SGFRI	SI3GT	C7C5	S	CNPR	Global Synch.	BSS/SS	See <i>TPF System Installation Support Reference</i>

Table 16. System-Required Fixed File Record Types (continued)

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#SONRI	CY3DR	C3C4	L	Pool Gen	File Pools	BSS/SS	
#SONSKP	CY7PL	C3C4	L	Pool	File Pools	BSS/SS	
#SORID	SS0OR	E2E2	L	CSOR	Non-SNA Comms.	SS	
#SPARI	SP0PA	E2D7	L/S	Online	SNA Comms.	BSS/SS	
#SRTRU	SR0RT	E2F8	4KB	Online	SNA Comms.	BSS	
#STCCR	CY\$CR	FF12	4KB	Pool Gen	Short Term Pool	BSS/SS	
#STPKP	CY3DR	C3C4	L	Pool	Maint. Short Term Pool	BSS/SS	
#STPUR	ICY\$PR	FF13	4KB	Pool Cycle Up	Short-Term Pool	BSS/SS	
#TDTDR	ITDAT	00FB	4KB	Restart	Tape Support	BSS	
#TER0 1-10	AS0MP	D4E2	L	ASL	Mapping Paging	BSS/SS	
#TPLBL	TAPEQ	E3E9	4KB	Restart	Tape Support	BSS/SS	
#UATRI	UA1UA	E4C1	L	STC	Non-SNA Comms.	BSS/SS	
#UQ1UQ	UQ1UQ	E4D8	S	STC	Non-SNA Comms.	BSS	
#UT2RT	UT2RT	E4E3	S	STC	Message Switching	BSS	
#UV1RP	UV1RP	E4E5	S	STC	Message Switching	BSS	
#UX1DQ	UX1DQ	E4E7	S	STC	Message Switching	BSS	
#VFARD	VF0AC	E5C1	L	VFA	VFA Support	BSS/SS	
#WAARI	WA0AA	C1C1	L	STC	Non-SNA Comms.	BSS/SS	
#WGTRU	WG0TA	C1D6	4KB	WGTA Table	Terminal Address Subtables	BSS	Required for the BSS only
#XCPYS	UQ1UQ	E4D8	S	Message Switching Restart	WTC	BSS/SS	
#XD0LS	XD0LS	E7C4	S	STC	Message Switching	BSS	
#XI1LD	XI0DS	E7C9	S	STC	Message Switching	BSS	
#XO1LD	XL0DS	E7D3	S	STC	Message Switching	BSS	
#XPRG1	IDSPRG	00FF	4KB	ACPL	C Language	BSS/SS	#XPRG is required.
#XQ1RI	XQ1XQ	E7D8	S	STC	Message Switching	BSS	
#XSQC	XY0XY	E7E8	L	STC	Message Switching and RES0	BSS	
#XS0RI	XS0AA	E7E2	S	STC	Non-SNA Comms.	BSS	

Table 16. System-Required Fixed File Record Types (continued)

Record Type	Record Macro Name	RIAT ID	Record Size	Initialized By	Functional Area	MDBF	Comments
#XS1AT	XB0XB	E7C2	S	STC	Message Switching	BSS	
#XT0RI	XT0CB	E7E3	S	STC	Message Switching	BSS	
#XV1XV	XV1XV	E7E5	S	STC	Message Switching	BSS	
#XZ1AT	XZ1AT	E7E9	S	STC	Message Switching	BSS	
<b>Notes:</b> 1. This record is processor shared. In TPF mail configuration file /etc/tpf_mail.conf, define a different #MAILxx record type for each domain in your mail system, where xx is a 2-character alphanumeric string, and allocate 1000 #MAILxx records for each domain that you define. For example, you can associate #MAIL01 with the mail1.site.com domain and #MAIL02 with the mail2.site.com domain. See <i>TPF Transmission Control Protocol/Internet Protocol</i> for more information about TPF Internet mail server support.							

Miscellaneous record types exist because the file address compute (FACE) program cannot allocate a single record of a specified type. Applications that require a single record should use a record from the appropriate miscellaneous record section. SIP RAMFIL macros for all three record types with a sufficient number of records are required. System required miscellaneous records use record types #IBMMS, #IBMML, #IBMMP4, and #IBMM4. See SYSEQ for more information about the contents of #IBMMS, #IBMML, #IBMMP4, and #IBMM4.

The terms PARS and IPARS, which appear in the Area column stand for the original applications that run under TPF and are called the Passenger Airline Reservation System and the International Passenger Airline Reservation System. A number of the miscellaneous records are only required for users of one or the other of these applications. In addition, World Trade users should refer to the appropriate IPARS version of the referenced documents. The *Fare Quote* package referred to is the IBM program product that was added to the original PARS application.

Table 17 contains the system record types located in the global area. These records are allocated as fixed file record types of #GLOBL. They are main storage resident with a backup copy on disk. Users should add their global area records to this list. A SIP RAMFIL macro for the #GLOBL record type is required. In addition, 2 or more records must be included in the #GLOBL area for the core allocator record (see *TPF System Installation Support Reference* for more information about global areas). Table 18 on page 111 contains all of the records checked by the FACE table generator. Table 19 on page 113 contains all of the records checked by the SIP Stage I assembly.

Table 17. Global Area Record Types

Macro Name	ID	Size	Initialized By	Area	Comments
AR0RT	AR	L	STC	Application Recovery	
AS2MT	MD	L	Online	Mapping	
XX1ON	XX	L	STC	Non-SNA Comm.	
GLOBB	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBC	GL	L	STC	Global Blocks	see INSL-GLBL

Table 17. Global Area Record Types (continued)

Macro Name	ID	Size	Initialized By	Area	Comments
GLOBD	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBE	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBF	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBG	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBP	GL	L	STC	Global Blocks	see INSL-GLBL
GLOBQ	GL	L	STC	Global Blocks	see INSL-GLBL
XC1CC	XC	L	STC	Message Switching	
XK1CT	XK	L	STC	Message Switching	
XF1FF	XF	L	STC	Message Switching	
XN1XN	XN	L	STC	Message Switching	
XJ1LC	XJ	L	STC	Message Switching	
XE1SC	XE	L	STC	Message Switching	
XW1OC	XW	L	STC	Message Switching	
XP1XP	XP	L	STC	Message Switching	
XA1DS	XA	L	STC	Message Switching	
XU2TQ	XU	S	STC	Message Switching	
XR1TR	XR	L	STC	Message Switching	

Table 18. Records Checked by the FACE Table Generator (FCTBG)

Recid	Size	Recno	Uniqueness	SS	Device	Required
#APRG1 to #APRG8	4KB	See note 38 See note 39	Shared	All	Any	No
#BKDRI	N/A	36	Shared	All	Any	Yes
#BREATB8	4KB	129	Shared	All	Any	Yes
#BRHIST8	4KB	1290	SSU	All	Any	Yes
#BRIDCOR	4KB	256	Shared	All	Any	Yes
#BRIDDE8	4KB	129	Shared	All	Any	Yes
#BRIDSA8	4KB	129	SSU and processor	All	Any	Yes
#BRIDTB8	4KB	129	SSU and processor	All	Any	Yes
#BRIDTO8	4KB	129	SSU	All	Any	Yes
#BRLOTB8	4KB	129	Shared	All	Any	Yes
#CCBRU	N/A	N/A	Processor	BSS	Any	No
#CIMR1	4KB	See note 23	Shared	All	Dev A	Yes
#CIMR2-8	4KB	See note 23	Shared	All	Dev A	No
#CN1ST	Small	64	Shared	BSS	Any	Yes
#CO1RI	N/A	N/A	Shared	BSS	Any	Yes
#CTKX	4KB	See note 24	Shared	All	Dev A	Yes
#CY2CPY	4KB	48	Shared	All	Any	Yes
#CY2KT	4KB	48	Shared	All	Any	Yes

Table 18. Records Checked by the FACE Table Generator (FCTBG) (continued)

Recid	Size	Recno	Uniqueness	SS	Device	Required
#CY2NEW	4KB	48	Shared	All	Any	Yes
#DBRRI	4KB	See note 1	SSU	All	Any	Yes
#DSCRUI	Large	See note 2	Processor	All	Any	Yes
#FLOCK	4KB	See note 32	Shared	All	Any	Yes
#GLOBL	Large	N/A	SSU	All	Any	Yes
#GR0ZSR	Large	See note 36	Shared	All	Any	Yes
#IBMML	Large	40	Shared	All	Any	Yes
#IBMMP4	4KB	100	Processor	All	Any	Yes
#IBMMS	Small	60	Shared	All	Any	Yes
#IBMM4	4KB	200	Shared	All	Any	Yes
#IDCF1	4KB	128	Shared	All	Any	No
#IDOTX	4KB	See note 3	Shared	BSS	Any	Yes
#INODE	4KB	See note 32	Shared	All	Any	Yes
#IPL1	4KB	See note 25	Shared	BSS	Dev A	Yes
#IPL2–IPL4	4KB	See note 25	Shared	BSS	Dev A	No
#IPRTE	4KB	N/A	Processor	BSS	Any	No
#IPSFB	4KB	See note 34	Shared	All	Any	No
#IZERO	4KB	See note 32	Shared	All	Any	Yes
#KBA	4KB	See note 4	Shared	All	Any	Yes
#KEYPT	4KB	See note 5	Shared	All	Dev A	Yes
#KFBX0 – 254	4KB	See note 6	Shared	BSS	Dev A	No
#KSA1	4KB	See note 4	Shared	All	Any	Yes
#KSA2 to #KSA8	4KB	See note 4	Shared	All	Any	No
#MPRECP	4KB	8	Shared	All	Any	Yes
#MQICD	4KB	51	N/A	BSS	Any	No
#NCBRI	Small	N/A	Shared	BSS	Any	No
#OLDx	4KB	See note 26	Shared	All	Any	No
#OSIT	4KB	See note 37	Shared	BSS	Any	No
#PROG1	4KB	See note 27	Shared	All	Any	Yes
#PROG2 to #PROG8	4KB	See notes 27 and 29	Shared	All	Any	No
#PSTXCUR	4KB	See note 40	Shared	All	Any	Yes
#PSTXNEW	4KB	See note 40	Shared	All	Any	Yes
#PVR1	4KB	See note 30	Shared	All	Any	Yes
#PVR2 to #PVR8	4KB	See notes 29 and 30	Shared	All	Any	No
#RCBRA	N/A	See note 7	Shared	All	Any	Yes
#RC8RFS	Large	257	Shared	All	Any	Yes
#RGSTAT	4KB	101	Shared	All	Any	Yes
#RLOG1–32	4KB	See note 33	Shared	BSS	Any	Yes
#RRTRI	4KB	N/A	Shared	BSS	Any	No
#RT1RI	4KB	See note 31	Processor	BSS	Any	No



Table 18. Records Checked by the FACE Table Generator (FCTBG) (continued)

Recid	Size	Recno	Uniqueness	SS	Device	Required
#RT2RI	4KB	See note 31	Processor	BSS	Any	No
#RV1RU	4KB	N/A	Processor	BSS	Any	No
#RV2RU	4KB	N/A	Processor	BSS	Any	No
#SATRU	Large	N/A	Processor	BSS	Any	No
#SC1RU	4KB	N/A	Processor	BSS	Any	No
#SC2RU	4KB	N/A	Processor	BSS	Any	No
#SONCP	Large	See note 35	Shared	All	Any	Yes
#SONDE	Large	See note 35	Shared	All	Any	Yes
#SONRI	Large	See note 8	Shared	All	Any	Yes
#SONROLL	Large	See note 35	Shared	All	Any	Yes
#SONRPE0–7	Large	See notes 9 and 35	Shared	All	Any	Yes
#SONRPM	Large	See note 35	Shared	All	Any	Yes
#SONSKP	Large	97	Processor	All	Any	Yes
#SONSV	Large	See note 35	Shared	All	Any	Yes
#SONUP	Large	See note 35	Shared	All	Any	Yes
#SPARI	N/A	N/A	Shared	BSS	Any	No
#SRHH1P	4KB	10	Shared	All	Any	Yes
#SRMP1A	4KB	10	Shared	All	Any	Yes
#SRM31A8	4KB	10	Shared	All	Any	Yes
#SRM41A8	4KB	531	Shared	All	Any	Yes
#SRM51A8	4KB	531	Shared	All	Any	Yes
#SRM61A8	4KB	10	Shared	All	Any	Yes
#SRTRU	4K	N/A	Processor	BSS	Any	No
#STCCR	4KB	See note 10	Shared	All	Any	Yes
#STPKP	Large	See note 35	Shared	All	Any	Yes
#STPUR	4KB	See note 11	Processor	All	Any	Yes
#TPLBL	4KB	See note 12	SSU and processor	All	Any	Yes
#UATRI	N/A	N/A	Shared	All	Any	Yes
#VFARD	N/A	See note 13	Shared	All	Any	Yes
#XPRG1	4KB	See note 28	Shared	All	Any	Yes
#XPRG2 to #XPRG8	4KB	See notes 28 and 29	Shared	All	Any	No
#XS0RI	N/A	N/A	Shared	BSS	Any	No

Table 19. Records Checked during SIP Stage I Assembly. All SIP Stage I records go through FCTBG and SIP checking; some attributes are checked by SIP Stage I and some are checked by the FCTBG.

Recid	Size	Recno	Uniqueness	SS	Device	Required or Required with
#ASMSI	N/A	36	Shared	All	Any	Mapping Support
#CANRI	N/A	N/A	Shared	All	Any	Reservation Support
#CCEPS	N/A	N/A	Shared	All	Any	3705 Support

Table 19. Records Checked during SIP Stage I Assembly (continued). All SIP Stage I records go through FCTBG and SIP checking; some attributes are checked by SIP Stage I and some are checked by the FCTBG.

Recid	Size	Recno	Uniqueness	SS	Device	Required or Required with
#GR0ZSR	N/A	N/A	Shared	All	Any	TPPDF Support
#GR3MSR	N/A	N/A	Shared	All	Any	TPPDF Support
N/A	N/A	Shared	All	Any	TPPDF Support	
#HDREC	4KB	15	Shared	BSS	Any	MPIF
#IDFCL	N/A	N/A	Shared	All	Any	TPPDF Support
#IDFUL	N/A	N/A	SSU	All	Any	TPPDF Support
#IDFCS	N/A	N/A	Shared	All	Any	TPPDF Support
#IDFC4	N/A	N/A	Shared	All	Any	TPPDF Support
#IDFUS	N/A	N/A	SSU	All	Any	TPPDF Support
#IDFU4	N/A	N/A	SSU	All	Any	TPPDF Support
#IRCBDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCFDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCGDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCHDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCIDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCJDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRCKDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IRDIDF	N/A	N/A	Shared	All	Any	TPPDF Support
#IR02DF	N/A	N/A	Shared	All	Any	TPPDF Support
#IR03DF	N/A	N/A	Shared	All	Any	TPPDF Support
#LKIBR	N/A	See note 14	Shared	All	Any	SLC Support
#NCBN4	4KB	N/A	Shared	BSS	Any	DLU Support
#NCBN5	4KB	N/A	Shared	BSS	Any	DLU Support
#PDREC	4KB	4	Shared	BSS	Any	MPIF Support
#PDREU	4KB	See note 15	Processor	BSS	Any	MPIF Support
#PTKST	N/A	See note 16	Shared	All	Any	2703 EP Support
#PTSCK	N/A	See note 17	Shared	All	Any	SCK Symbolic Lines
#RCATU	L	See note 18	Processor	BSS	Any	Yes
#SGFRI	N/A	See note 19	SSU	All	Any	Yes
#TDTDR	4KB	See note 20	Processor	BSS	Any	Yes
#TER0x	N/A	N/A	Shared	All	Any	Mapping Support
#UQ1UQ	N/A	N/A	Shared	BSS	Any	Non WTC Support
#WAARI	N/A	See note 21	Shared	All	Any	Reservation Support
#WGTRU	4KB	See note 22	Processor	BSS	Any	Yes
#XCPYS	N/A	N/A	Shared	All	Any	WTC Support
#XSQC	N/A	64	Shared	BSS	Any	Reservation Support or Message Switching Support for non-WTC

These notes apply to both the records checked by the FACE table generator and those checked during Stage I.

**Notes:**

1. Number of record types + 3
2. For Large record size, calculate as follows:

Number\_of\_general\_data\_sets/8  
with a minimum of  $2 \times \text{number\_of\_processors}$

For 4KB record size, calculate as follows:

Number\_of\_general\_data\_sets/31  
with a minimum of  $2 \times \text{number\_of\_processors}$

The maximum number of records is 255 for each processor.

3.  $24 \times \text{number of processors}$
4. Calculated as follows:

$((\text{number\_of\_processor-unique\_keypoints} \times \text{number\_of\_processors}) + \text{number\_of\_processor-shared\_keypoints} + 1)$

5. This is a vertical record. Multiple extents (RAMFIL macro with PRIOR parameter value not equal to 0) are allowed.

$((\text{number\_of\_processor-unique\_keypoints} \times \text{number\_of\_processors}) + \text{number\_of\_processor-shared\_keypoints}) \times (\text{NFBACK\_parameter\_on\_RAM\_macro} + 1) + 1$

6. This is a vertical record that must be defined if the value of the NFBACK parameter on the RAM macro is not 0. The number of definitions must match the value of the NFBACK parameter and must start with #KFBX0 and continue sequentially.

$(1 + (7 \times \text{number\_of\_processors}) + 9)$

7.  $2 \times \text{number of processors}$
8. Add up for each pool  $\text{RAMFIL} (\text{RECNO} + 7999)/8000$ .
9. Define #SONRPE<sub>x</sub> records for each processor up to the first eight. If there is one processor, only define #SONRPE0. For additional processors, define #SONRPE1 for processor 2 up to SONRPE7 for processor 8. Do not define additional SONRPE<sub>x</sub> records for processors beyond the eighth.
10.  $12 + \text{the sum of all } ((\text{short-term pool directories} + 499) / 500)$
11. This is a processor unique record.

The number of records for each processor =  
 $12 + (((\text{number of short-term directories for SST-A} + 499)/500)$   
 $+ ((\text{number of short-term directories for LST-A} + 499)/500)$   
 $+ ((\text{number of short-term directories for 4ST-A} + 499)/500)$   
 $+ \dots$   
 $+ ((\text{number of short-term directories for 4ST-D} + 499)/500))$

**Notes:**

- a. Drop the fractional portion of each short-term directory result, for example:  
 $(7 + 499)/500 = 1$ .
- b. All 12 pool short-term directory types (SST-A to 4ST-D) must be included in the calculation.

12. This is an SSU/processor unique record. (18 ordinals for each RAMFIL statement.).
13. Number of processors
14. Number of pool records per link × number of SLC links
15. Calculated as follows:  
  

$$\frac{\text{number of paths defined for the TPF system/}}{((4096 - \text{record header length})/\text{PDR item length})}$$
16. Number of 2703 lines
17. Number of symbolic lines
18. Number of RCAT records
19. Number of fields defined for the user in GLSYNC. One more than this is required for the first SSU.
20. This is a processor unique record. (Number of tape control units + 1 ordinals) for each RAMFIL statement.
21. Number of unique LNIATAs
22. Calculated as follows:  
  

$$\begin{aligned} &(((\text{number of UATs} \times \text{WGTA entry size} + \\ &\quad (\text{calculated usable space of a 4K block} - 1))/ \\ &\quad (\text{calculated usable space of a 4K block})) + \\ &(((\text{number of UATs} \times \text{hash table entry size} + \\ &\quad (\text{calculated usable space of a 4K block} - 1))/ \\ &\quad (\text{calculated usable space of a 4K block}))) \end{aligned}$$
23. This is a vertical record. The RECNO parameter on the RAMFIL macro that is coded for this record type reflects the records that are reserved on a single module only. The total number of records, which is the RECNO times the number of prime modules, must be enough to contain all the CIMR components. All RECNO must be coded with a single RAMFIL. #CIMRx cannot be spanned.
24. This is a vertical record. The RECNO parameter on the RAMFIL macro that is coded for this record type reflects the records that are reserved on a single module only. The total number of records that are coded for this record type must equal the number of images to be generated. If only 1 image is being generated, RECNO=2 must be coded. #CTKX cannot be spanned.
25. This is a vertical record. The RECNO parameter on the RAMFIL macro that is coded for this record type reflects the records that are reserved on a single module only. The total number of records coded for this record type must be large enough to contain IPL1, IPL2, IPLA, and IPLB. #IPLx cannot be spanned.
26. RECNO must be greater than (maximum number of loadsets online × average number of programs in each loadset) × 1.1  
  
For the BSS subsystem, #OLDx records are required; the minimum number allowable is defined by the following equation. For other subsystems, users can choose not to allocate any #OLDx records if they do not plan to use the E-type loader on that subsystem.

Minimum number of #OLDx records required =

$$\begin{aligned} &35 \quad + \quad (\text{\#CPUs genned} \times 200) \quad + \\ &(\text{reserved}) \quad \quad \quad (\text{recommended number of} \\ &\quad \quad \quad \text{available records}) \end{aligned}$$

$$\begin{array}{lll} (32 & \times & [\text{MAX\_NUM\_ACTIVE\_LOADSETS} / \text{ECR\_NUM\_ENTRIES}]) \\ (\text{Max.} & & (\text{IBM default} = 1000) \quad (\text{Defined in} \\ \text{CPUs}) & & \text{C\$IDSECR} = 203) \end{array}$$

Round up the figure in square brackets to a whole number.

You may need to define additional #OLDx records because C load modules use a minimum of 2 #OLDx records each and can be larger than 4 KB. Use the following formula to estimate the number of additional #OLDx records that you need. The formula can be used in either of the following ways:

- On a per C load module basis and then summed up for a total.
- Use the total size of all C load modules for the total number of #OLDx records needed.

This needs to be calculated only for the maximum number of C load modules that will exist in loadsets at a given time. The size of the C load module (size\_of\_module) is the offline size produced by the linkage editor. Each directory record can hold approximately 500 entries.

The following formula is an approximation; it is not intended to give an exact number. It can be used on from 1 to  $n$  number of C load modules.

$$\begin{array}{l} \text{Number of \#OLDx records for one or more C load modules} = \\ ((\text{sum\_of\_C load module\_sizes}/4000) + (3 * \text{number\_of\_C load modules})) \end{array}$$

Some examples help to explain this formula.

- A C load module that is 1 MB in size:

$$\begin{array}{rcl} \text{Number of \#OLDx records} & = & ((1000000/4000) + (3 * 1)) \\ & = & 250 + 3 \\ & = & 253 \end{array}$$

- A C load module that is 1.624 KB in size:

$$\begin{array}{rcl} \text{Number of \#OLDx records} & = & ((1624/4000) + (3 * 1)) \\ & = & 0.406 + 3 \\ & = & 3.4 \\ & = & 4 \end{array}$$

- A loadset that contains 50 C load modules, 15 of which contain multiple OBJs, and 35 that contain only one OBJ each, for a total of 289.352 KB in size:

$$\begin{array}{rcl} \text{Number of \#OLDx records} & = & ((289352/4000) + (3 * 50)) \\ & = & 72.338 + 150 \\ & = & 222.338 \\ & = & 223 \end{array}$$

You will need to allocate additional #OLDx records if you plan to use the E-type loader to load programs with ADATA files for use by the TPF Assembler Debugger for VisualAge Client. To estimate the number of additional #OLDx records that need to be allocated, do the following:

- Estimate the maximum number of basic assembler language (BAL) programs that will be loaded with ADATA files in loadsets simultaneously.
- Calculate the number of records required to hold ADATA files by using one of the following:

Number of #OLDx records for BAL ADATA files = (number of BAL programs)  
+ ((sum of sizes of object modules) / 80)

Number of #OLDx records for BAL ADATA files = (number of BAL programs)  
+ ((sum of sizes of source files) / 1200)

Number of #OLDx records for BAL ADATA files = (number of BAL programs)  
\* 40

- c. Modify the number of #OLDx records for BAL ADATA files by using one of the following:

- If the number of BAL programs is less than 10, multiply the number of #OLDx records by 1.5.
- If the number of BAL programs is greater than 10 but less than 25, multiply the number of #OLDx records by 1.3.
- If the number of BAL programs is greater than 25 but less than 50, multiply the number of #OLDx records by 1.1.

- d. Estimate the maximum number of assembler programs that are members of load modules that will be loaded with ADATA files in loadsets simultaneously.

- e. Calculate the number of records required to hold ADATA files by using one of the the following:

Number of #OLDx records for load module ADATA files = (2 \*  
(number of load modules containing members with ADATA files)) +  
((sum of sizes of object modules) / 80)

Number of #OLDx records for load module ADATA files = (2 \*  
(number of load modules containing members with ADATA files)) +  
((sum of sizes of source files) / 1200)

- f. Modify the number of #OLDx records by using one of the following:

- If the number of load module members is less than 10, multiply the number of #OLDx records by 1.5.
- If the number of load module members is greater than 10 but less than 25, multiply the number of #OLDx records by 1.3.
- If the number of load module members is greater than 25 but less than 50, multiply the number of #OLDx records by 1.1.

- g. Add the number of #OLDx records for BAL ADATA files (calculated in the previous steps) to the number of #OLDx records for load module ADATA files (calculated in the previous steps) to calculate the total number of additional #OLDx records required to hold ADATA files.

**Note:** These guidelines provide only a rough estimate, so increase or decrease the allocations based on available space and the importance of having enough records available for loadsets.

27. RECNO must be greater than or equal to the total number of programs, transfer vectors, and spares to be loaded.
28. The formula for determining the number of #XPRGns can be used in either of the following ways:
- On a per C load module basis and then summed up for a total.
  - Use the total size of all C load modules for the total number of #XPRGn records needed.

The size of the C load module is needed as input to the formula. It is the offline size produced by the linkage editor.

The following formula is an approximation; it is not intended to give an exact number. It can be used on from 1 to  $n$  number of C load modules.

Number of #XPRGn records for one or more C load modules =  
 $((\text{sum\_of\_C load module\_sizes}/4000) + (2 * \text{number\_of\_C load modules}))$

Some examples help to explain this formula.

a. A C load module that contains multiple OBJs and is 85.296 KB in size:

$$\begin{aligned} \text{Number of \#XPRGn records} &= ((85296/4000) + (2 * 1)) \\ &= 21.324 + 2 \\ &= 23.324 \\ &= 24 \end{aligned}$$

b. A C load module that is 1.624 KB in size:

$$\begin{aligned} \text{Number of \#XPRGn records} &= ((1624/4000) + (2 * 1)) \\ &= 0.406 + 2 \\ &= 2.406 \\ &= 3 \end{aligned}$$

c. Fifty (50) C load modules, 15 of which contain multiple OBJs, and 35 that contain only one OBJ each, for a total of 289.352 KB in size:

$$\begin{aligned} \text{Number of \#XPRGn records} &= ((289352/4000) + (2 * 50)) \\ &= 72.338 + 100 \\ &= 172.338 \\ &= 173 \end{aligned}$$

29. For any additional program base, you must have a corresponding set of #PROGn, #PVRn, and #XPRGn records. For example, #PROG2 must have corresponding #PVR2 and #XPRG2 records. You cannot have 1 of these record types without both of the other 2 record types.
30. The formula for estimating the number of #PVR records for 1 program base is:  
 Number of #PVRn records = (number of #PROGn records) / PVRMAXE  
 for a program base

Note that PVRMAXE is defined in IDSPVR; the current value of PVRMAXE is 127.

31. This is a processor unique record. See *TPF ACF/SNA Network Generation* for information about how many #RT1RI and #RT2RI records to define.
32. You must have equal numbers of #INODE and #FLOCK fixed file records. The number of #INODE and #FLOCK fixed file records must be allocated in fixed ratios: up to 1001 #INODE/#FLOCK records must have 1 #IZERO fixed file record, up to 2001 #INODE/#FLOCK records must have 2 #IZERO fixed file records, up to 3001 #INODE/#FLOCK records must have 3 #IZERO fixed file records, and so on.
33. This is a vertical record. The RECNO parameter on the RAMFIL macro that is coded for this record type reflects the records that are reserved on a single module only.

**Note:** For TPF transaction services, the recovery log fixed file records are normally allocated to the BSS. If you do not want to allocate to the BSS, you may allocate recovery log records to any user-defined application subsystem. You must always allocate recovery log records on the BSS even when they have been allocated on the user-defined application subsystem. BSS recovery log records are necessary if the system is IPLed without the user-defined application subsystem. The TPF system must always have access to recovery log records.



To calculate the required number of records, use the following formula:

$$(300 \text{ ECBs per second} * \text{the average size of the commit scope for each ECB} \\ (100 \text{ KB}) * \text{TPF queue manager checkpoint interval}(5 \text{ seconds})) + \\ (1000 \text{ MQPUT/MQGET requests per second} * \text{the average size of a message} \\ (4 \text{ KB}) * \text{TPF queue manager checkpoint interval}(5 \text{ seconds})) * 1.5 \\ (\text{factor for padding})$$

For example,  $(300 * 100 \text{ KB} * 5 = 150\,000 \text{ KB}) + (1000 * 4 \text{ KB} * 5 = 20\,000 \text{ KB}) * 1.5 = 255 \text{ MB}$ .

34. #IPSFB fixed file records (RIAT ID X'FC40') are used in a circular fashion. Ordinal record 0 is used to control the use of the remaining records. Ordinal records 1 through the number of records that were defined will have an entry added to them for each program that is activated, deactivated, accepted, or changed (using the ZAPAT, ZAPGM, or ZRTDM commands). The size of the database is dynamically allocated based on the number of #IPSFB records that are defined. The CUPT program logs the entries. For example, if you activate a loadset with 50 programs, positive feedback support makes 50 entries to the fixed file record starting at the 51st entry in the current 4-KB record. When ordinal record 1 is filled, positive feedback support makes entries in ordinal record 2. Positive feedback support fills each record until the last record that was defined. After making an entry in the last slot in the last record, positive feedback support makes its next entry in the first slot of the first record, ordinal 1. Therefore, the number of #IPSFB fixed file records that are defined, as well as the amount of loggable activity, determines how much information can be stored before the log wraps.

When the TPF system is cycled to NORM state, the CUPH program copies the contents of the #IPSFB fixed file records to a POSIX file called `tpfva.fil`, which is found in directory `/tpf.va`. This file is a duplicate of the fixed file records. It can be transferred to another system by using the TPF-supported Trivial File Transfer Protocol (TFTP) or File Transfer Protocol (FTP). When the file has been transferred, it can be used by any user program.

To turn off positive feedback support, remove the #IPSFB records.

35. Greater of (the number of ordinals allocated for SONRI) or the sum of each pool `RAMFIL (RECNO + 7999)/8000`
36.  $30 \times \text{number of processors}$
37. Minimum of  $4 \times \text{number of processors}$ . Define #OSIT records only if Open Systems Adapter (OSA)-Express Support is defined (the MAXOSA parameter is not zero).
38. #APRGx records are needed for any program base on which you plan to load ADATA files with base versions of programs. To estimate the number of additional #APRGx records that need to be allocated, do the following:
  - a. Estimate the maximum number of BAL programs that will be loaded with ADATA files in loadsets simultaneously.
  - b. Calculate the number of records required to hold ADATA files by using one of the following:
$$\text{Number of \#APRGx records for BAL ADATA files} = (\text{number of BAL programs}) \\ + ((\text{sum of sizes of object modules}) / 80)$$
$$\text{Number of \#APRGx records for BAL ADATA files} = (\text{number of BAL programs}) \\ + ((\text{sum of sizes of source files}) / 1600)$$
$$\text{Number of \#APRGx records for BAL ADATA files} = (\text{number of BAL programs}) \\ * 40$$
  - c. Modify the number of #APRGx records for BAL ADATA files by using one of the following:



- If the number of BAL programs is less than 10, multiply the number of #APRGx records by 1.5.
  - If the number of BAL programs is greater than 10 but less than 25, multiply the number of #APRGx records by 1.3.
  - If the number of BAL programs is greater than 25 but less than 50, multiply the number of #APRGx records by 1.1.
- d. Estimate the maximum number of assembler programs that are members of load modules that will be loaded with ADATA files in loadsets simultaneously.
- e. Calculate the number of records required to hold ADATA files by using one of the following:
- Number of #APRGx records for load module ADATA files = (2 \*  
(number of load modules containing members with ADATA files)) +  
((sum of sizes of object modules) / 80)
- Number of #APRGx records for load module ADATA files = (2 \*  
(number of load modules containing members with ADATA files)) +  
((sum of sizes of source files) / 1200)
- f. Modify the number of #APRGx records by using one of the following:
- If the number of load module members is less than 10, multiply the number of #APRGx records by 1.5.
  - If the number of load module members is greater than 10 but less than 25, multiply the number of #APRGx records by 1.3.
  - If the number of load module members is greater than 25 but less than 50, multiply the number of #APRGx records by 1.1.
- g. Add the number of #APRGx records for BAL ADATA files (calculated in the previous steps) to the number of #APRGx records for load module ADATA files (calculated in the previous steps) to calculate the total number of additional #APRGx records required to hold ADATA files.

**Note:** These guidelines provide only a rough estimate, so increase or decrease the allocations based on available space and the importance of having enough records available for ADATA files.

39. To use expression enhancements for the TPF debuggers, you must double your APRGx record allocation. Symbol tables are created by the TPFSYM offline program and contain information for all symbols defined in a program. The TPFSYM offline program is loaded to the TPF system through the TPFLDR offline loader. The increased size of each ADATA file depends on the number of symbols defined in the program and the number of symbols defined in the common symbol table (UCST). Doubling your APRGx record allocation provides enough room to hold the new ADATA file.
40. Minimum of 10 ordinals. If more than 710 pool RAMFIL statements exist then the number of ordinals is the (number of pool RAMFIL statements + 70)/71.

---

## Standard Data/Message File

As part of the released system, there is a file included known as the standard data/message file (SDMF). This file is associated with the offline running of the system test compiler (STC) and consists of two parts, both of which will need to be expanded by the user.

The first part is known as the data record information library (DRIL) and is used to symbolically define all data records to be created using STC. The released version

of this file contains the symbolic definition of all system-related data records. Users should add to DRIL the definition of their own application data records.

The second file is known as SDMF and is similar to DRIL but in addition it allows the records to be initialized with information contained within the SDMF file. Individual fields within the data record can be initialized uniquely. In addition to defining his records in DRIL the user may wish to add his new records to SDMF. For example, as a minimum the user may initialize the record headers (e.g., the record ID) using this facility.

More information about both SDMF and DRIL can be found in the *TPF Program Development Support Reference*. All of the system-related data records, which have been previously mentioned in this document, are defined in DRIL and many are also initialized via SDMF.

---

# Data Communications Support

---

## Basic TPF Communications Support

The communications lines that are supported by the TPF system may be divided into 3 categories: SNA, non-SNA, and communications console support.

- SNA
  - Synchronous Data Link Control (SDLC)

Users can implement TPF's Advanced Communications Function (ACF), which is required for TPF's implementation of SNA and extends the basic support provided for SDLC as well as providing support for ALC terminals connected through ALCI. Considerations for this support are described in *TPF ACF/SNA Network Generation* and *TPF ACF/SNA Data Communications Reference*.
- Non-SNA
  - ATA-IATA Synchronous Line Control (SLC)

**Note:** ATA-IATA is the Air Transport Association and the International Air Transport Association, which set the standard for this protocol.
  - Binary Synchronous Communications (BSC)
  - 3270 Local (LC)

These 3 protocols are known as non-SNA communications support. SLC is associated with lines that are connected to a transmission control unit (TCU) such as the 3705. The 3705 itself is run under the control of the 3705 Emulator Program (3705/EP) and can be attached to the host processor executing TPF only on byte multiplexor channel 0. BSC runs on a 3745 loaded with the partitioned emulator program (PEP). 3270 local control units may execute TPF on either a byte multiplexor channel or a block multiplexor, attached to logical channel 0.

- Communications Console Support (NCS or 1052/3215 )

**Note:** NCS is the TPF term used for *native console support* and implies support for the system operator console, which comes as part of the processor (for example, 3036, 3278 Model 2A, and so on). Support for the 1052/3215 typewriter (which is connected to the processor via a 7412 RPQed control unit) is also provided by the TPF system.

Users of the High Performance Option (HPO) feature also have communications considerations that must be addressed. They are introduced later in "High Performance Option Feature" on page 124.

This publication does not address the generation of 3705/EP, 3745/PEP, or ACF/NCP. The reader is directed to the appropriate publications for those programs for this aspect of system generation. Where necessary, reference will be made to certain aspects of this separate generation as it relates to the TPF generation.

Another general consideration that the system designer of the TPF communication support must be aware of is the relationship of the communications support, specifically each of the protocols supported, to the applications that will be written/employed on the TPF system. The TPF message router (see "Message Router Support" on page 142) allows you to define an application as either an old or new application. The connotation here is that non-SNA communications-supported

protocols interface with old applications, where the terminal address (for example, the line number, terminal interchange number) is used as an index number (an ordinal number) for a fixed file record (typically an agent assembly area (AAA)); whereas a new application does not employ this concept. A bridge technique is used for 3270/SDLC terminals to communicate with an old application via a pseudo terminal address.

Although TPF can support a system configuration having all of the previous types of lines, it is probable that only a limited number of these types would be used in a particular system. Each type of line can be used only with prescribed types and models of equipment such as terminals, concentrators, transmission control units, processors, and so on.

After your system designer has arrived at a final configuration showing the various types of lines, the number of lines in each type, and equipment such as transmission control units, controllers, concentrators, terminals, and so on that are attached to each line, you are ready to code certain sets of macros that allow for the generation and loading of the communication database.

---

## High Performance Option Feature

Both the loosely coupled facility (LC) and the multiple database function (MDBF) of TPF/HPO present unique communications considerations for users of either or both options, LC or MDBF.

## Loosely Coupled Communications Considerations

ACF support is a prerequisite for the use of LC because the individual processors of the LC complex are connected to one another by MSNF and the multitail channel connection feature of the 37x5 (in addition to the DASD connection provided by the ELLF RPQ on the DASD control units). This allows each processor in the LC complex direct access to all SDLC and NEF supported ALC devices. This use of MSNF and SNA networking is just an LC implementation technique that is allowed by the fact that TPF/ACF support already supports the connection of multiple TPF/ACF systems. The difference, is that the various TPF/ACF systems are also loosely coupled together via the ELLF RPQ. It should also be noted that an LC complex may also be connected via MSNF to other hosts and their communication networks, including other TPF/HPO LC complexes.

Normally, all but one of the LC processors can be active at one time with the other connected to the 37x5 acting as a backup processor in the event of a switchover from any of the other processors. Outside of any 37x5 hardware limitations, the HPO feature itself supports as many as 32 processors loosely coupled together.

The use of 3705 EP or 3745 PEP in an LC configuration presents a problem because support for a multitail connection to multiple hosts is not provided by the 3705 EP or the 3745 PEP. As a result, LC employs the concept of a single EP processor, which is located in the LC complex, to which is attached the TCUs running 3705 EP. This EP processor is also the only one that is allowed to have 3270 local control units and terminals attached to it except for native console support. This means that all applications or functions that are only associated with the non-SNA network must be associated with the EP processor. For example, support facilities such as message switching (MESW) (see "Message Switching Packages" on page 152) are also located in the EP processor.

System generation of communications support for LC is basically the same as generating a TPF system with ACF support with the exceptions noted later. Support for the LC facility itself is the result of the user installing the HPO feature and specifying multiple TPF processor IDs on the SYSID parameter of SIP's CONFIG macro. In addition to the generation procedures outlined later on in this section, users are also directed to several of the communications and system support packages that appear in "Application and System Support" on page 141, and especially "Message Router Support" on page 142.

## Multiple Database Function Communications Considerations

In a TPF/HPO system with MDBF, communication support is a shared resource with all communications support functions being located in the basic subsystem (BSS). As a result, the communications database is only generated for the BSS and is not relevant to the other subsystems. An exception to this rule is the AAA/RCB initialization table (UAT), which is generated as part of the communication pilot record process (see "Communication Pilot Record Generation" on page 137).

Terminals, which are defined by the UAT, are supported by the BSS, but at any given time are connected or logged to an application that is located in a particular subsystem user or subsystem. (See "Message Router Support" on page 142 and "Log Processor" on page 145.) As a result, there needs to be a UAT definition by subsystem for all terminals that can log to applications located in that subsystem. In theory, all subsystems could define all terminals, or a particular subsystem only has to define those terminals in its UAT that will log to it. The BSS itself needs a definition of all terminals on the system.

**Note:** All non-SNA terminals are defined in the UAT. For SNA devices only, 3270/SDLC and all NEF terminals, which communicate with old applications, need an entry in the UAT. This entry is the pseudo address that will be associated with the terminal when it is logged to an old application.

Support for MDBF itself, including communications support, comes as a result of a user coding SIP SSDEF macro. In addition, unique MDBF parameters are associated with other communications macros that will be noted as they are discussed.

---

## Communications Record Generation

Depending on the line types selected for the communications configuration, as many as 5 different sets of system generation macros or processes may have to be coded or executed in order to generate the required communications network records. In addition, users must code or execute a generation of 3705 EP, 3745 PEP, ACF/NCP or all, if they want any support other than the system console or 3270 local terminals. The 5 sets of macros or processes are as follows:

- System initialization package (SIP) communication macros described in "Stage I: Coding the SIP Macros" on page 207. Some of the SIP macros are required regardless of the line types and some are optional depending on line type.
- System communication keypoint (SCK GEN) macros and communication control unit keypoint status records (PKST) macros described in this publication (see "Non-SNA Communications" on page 335). These are required for all non-SNA networks except for the system operator console (prime CRAS) and for the system operator receive only printer (RO CRAS).

- System test compiler (STC) macros described in the *TPF Program Development Support Reference* are required for several communications-related records. These records are created on tape offline by STC and then introduced to the online system via pilot tapes.
- ACF/SNA Table Generation (OSTG) input definition statements described in *TPF ACF/SNA Network Generation*.
- The SNAKEY macro in the SNA communications keypoint (CTK2) must be updated to reflect the user's SNA network. CTK2 must then be assembled and included in the TPF load deck in order to activate the SNA support.

The remainder of this chapter is devoted to explaining the design considerations for these 5 generation macros and processes. The reader is directed to the previously referenced chapters of this publication for detailed information on how to code the various macros and how to execute the various procedures.

## SIP Communication Macros

Table 20 lists the various SIP macros related to communications and the associated records and macros modified and generated by SIP as a result of the user-specified macro parameters. Users should review each macro listed and code the macro parameters that are applicable to their communications network. There are a number of SIP macros and keywords that do not have to be coded if you want to accept the defined default values. Only code these macros for the BSS of MDBF users.

SIP will always generate the code necessary to support SNA communications in the generated TPF system. In order to activate this support, the user must update the SNAKEY macro in the SNA communications keypoint (CTK2) and reassemble and load the keypoint to the TPF system. See *TPF ACF/SNA Network Generation* for information on this part of SNA network definition.

Table 20. SIP Communications Macros

SIP Macros	Macros and Records Modified
BBSAT	BSAT, CTKE, SYCON, SYSETK
BSNCT	CRSx, JPC0, IBMPAL, SNCT, SYCON, SYSETK
CCPERR	LINEQC, SYCON, SYSETK
CCPPOL	CTKA
CCPSTA	CTKA
CRASTB	BSSSET, CTKC, LINEQC, SYSET, SYSTG
LINES	BSSSET, CTKA, CTKE, JPC0, IBMPAL, SYCON, SYSET, SYSETK
MSGRT	BSSSET, COHx, CTKA, CTKC, JPC0, IBMPAL, SPPGML, SYCON, SYSET, SYSTG
MSGRTA	ANTDEF, COHx, CRSx, JPC0, SYCON
NETWK	BSSSET, CC0CC, CC1CC, CTKA, CTKB, CTKC, CTKE, JPC0, LINEQ, SYCON, SYSET, SYSETK, SYSTG
SYNCLK	BSSSET, CTKA, CTKD, SYCON, JPC0, SYSET

Various communication functions are related to the SIP macro parameters in the following ways:

Error control data (CCPERR macro): This macro controls the error recovery processing done by the communications control program (CCP) for non-SNA

communications support. The error counts are threshold values which, when reached, will cause CCP error recovery and line shutdown to begin. The default values supplied by SIP have proven acceptable with past systems and most likely do not require modification. If users want to change these values they should consult the appropriate IBM hardware publication before doing so. A listing of error counts and their supplied defaults can be found in "CCPERR" on page 212.

Line shutdown and restart levels (CCPPOL macro): This macro establishes the input list sizes that stop and start the acceptance and processing of input from the various non-SNA line types. Two levels are supplied for each line type function, as follows:

- Shutdown level: When the number of items on the input list reaches this level, polling of that line type ceases. Note, all messages going to the processor from the last poll are processed.
- Restart level: At this input list size, normal polling is resumed.

**Note:** For 3270 locals that are not polled, the shutdown and restart levels apply to what action is taken by the host on the receipt of an attention interrupt.

The shutdown and restart levels default to an input list length of 300 for shutting down and an input list length of 150 for restarting.

Users should be aware of the number of times that line polling is shut down and tune their system accordingly. In order to evaluate the efficiency of the levels, refer to the system reduction report produced by the data collection and reduction package. This report will indicate the number of times during samples that lines were shut down and restarted.

All of the levels specified to SIP may be displayed or altered dynamically online via the ZLDCL/ZLACL command.

CTKA communication options (CCPSTA macro): Several communication system options may be set (or defaulted) in the CTKA keypoint record as follows:

- The CCP may be initialized (MPXID) with or without multiplexor lost-interrupt detection. This feature causes a program to be periodically activated, which detects whether or not an interrupt has been sensed on each subchannel expecting an interrupt. If, after 2 activations (approximately 2–4 minutes), no interrupt has been sensed on a given subchannel, an interrupt to that subchannel is forced. This interrupt frees the subchannel for more activity.
- The frequency of a CCP time-initiated keypoint update (KPUPD) must be specified to control the interval between updates.

SYCON macro communication variables (NETWK macro): Certain miscellaneous values required by SYCON may be set through SIP as follows:

- The maximum number of alternate paths (NPATH) through which a line can be connected to the system may be set to a value of 1 to 4. This value should be established with the number of line paths defined during SCK/PKST generation. (See "Communication Keypoint Macros (Non-SNA)" on page 132.)

37x5s attached to the TPF system may run in 3 modes depending on the program loaded into the 37x5. The 3 modes are:

- NCP—Network Control Program  
NCP mode supports only SNA protocols.
- EP—Emulation Program



EP mode supports only non-SNA protocols.

- PEP—Partitioned Emulation Program

PEP mode supports both SNA and non-SNA protocols.

**37x5 NCP mode:** The symbolic device address of the 37x5 is defined with the SIP IODEV macro. Further definition of the NCP and its resources is accomplished with the NCP and RSC input definition statements in OSTG. The 37x5 in NCP mode may not be IPLed by the TPF system. It must be owned and IPLed by the required VTAM CMC.

**37x5 PEP mode:** The symbolic device address that is associated with the NCP should be specified with an IODEV macro. See “NETWK” on page 279 for definitions necessary for the EP. Any NCP (for example, SNA) resources in the PEP are defined as NCP resources with the NCP and RSC input definition statements of OSTG. EP resources in the PEP are defined to TPF via SIP as if the 37x5 was an EP. A 37x5 running in PEP mode may **not** be IPLed by TPF. It must be owned and IPLed by the VTAM CMC.

**37x5 EP mode:** TPF supports the EP mode only for 3705. A 3705 in EP mode may be IPLed by the TPF system. For 3705s in EP mode, enter the following SIP input:

- The number (MAXDP) of 3705/EP dumps retained on file. This is the total number of 3705 dumps retained on file before logging to tape. If this number is exceeded, dumps are lost.
- The maximum number (MAXEP) of 3705/EP load modules allowed on file.
- The addresses and channel adapter types (SUBCH) of the attached 3705s running 3705/EP.
- The diagnostic support available (DIAG) for each of the 3705s running 3705/EP is not supported by TPF.
- The loosely coupled user of the HPO feature must also specify which processor in the LC complex will act as the *EP processor*. The EP processor will always be the first TPF processor ID coded on the SYSID parameter of the SIP CONFIG macro. (See “Loosely Coupled Communications Considerations” on page 124).

If the system being generated contains 3745 control units in ACF/NCP or PEP mode, additional network definition is required in OSTG and the SNAKEY macro in the SNA communications keypoint (CTK2). (See *TPF ACF/SNA Network Generation* for the specific definitions required.)

**Binary synchronous communications (BSC) inputs:** The BSC procedures as described in *General Information - Binary Synchronous Communications* are supported under TPF. The following SIP macros are required to be coded, as applicable, in order to initialize the BSC communication lines:

- LINES
- NETWK
- BBSAT
- BSNCT
- CCPERR
- CCPPOL

The NETWK macro parameters are shown in Table 22 on page 136, and the LINES macro is shown in Table 21 on page 134. These macros require separate entries for each line and TCU that utilizes BSC.



SCK/PKST communication keypoint macros (see “Communication Keypoint Macros (Non-SNA)” on page 132) are also required to be coded, as applicable per worksheet entries for both Table 21 on page 134 and Table 22 on page 136.

The BBSAT macro must be coded if there are any BSC multipoint lines directly connected to the TPF host processor. It is used to specify the valid sets of polling and selection characters in the BSC multipoint network. Each BBSAT macro relates to a unique BSC line that has also been defined in the SCK and 3745/PEP generations.

The BSNCT macro must also be coded for any BSC multipoint lines. It may also be optionally used for BSC point-to-point lines.

The BSC-associated keywords must also be coded on both the CCPERR and CCPPOL macros to provide for BSC error control and for the BSC restart or shutdown levels.

Synchronous link control (SLC) inputs (SYNCLK macro): If the system being designed contains SLC lines, several variable fields used in SYCON and keypoint D (CTKD) plus the SLC routing table (CTKD) must be defined through SIP. These values are in addition to the SLC lines and pseudo-lines (if any) that must also be described to SIP (LINES macro) and the SCK/PKST macros as shown in Table 16 on page 105.

The parameters that follow are all those entered in the SIP SYNCLK macro. A brief description accompanies each. A full description of SLC may be found in the *TPF Non-SNA Data Communications Reference*.

1. The number of SLC input pool records (NPOOL) must be defined. This number must be a multiple of 8.
2. The number of SLC links (NLINK) in the system is required if pseudo-lines are being used.
3. If there are SLC links in the system, indicate the number (N1LNK) with a single AI (ALC) line pair. This number must not exceed the number of SLC links.
4. The number of slots in the input link control block queue (ILCBQ) must be designated.
5. The number of slots in the output link control block queue (OLCBQ) must also be specified.
6. The AI line CXFRC/ENQ LCB-IN level (AILCB) must be established.
7. The routing of input messages must be established in 2 parameters. The first input (AILST) defines the queue on which the CCP will place input message blocks at initial entry time. The choice is between the ready list (higher priority) and the input list (lower priority). If the input list is chosen no additional coding is required. If the ready list is specified, the routing of the last received or only block of type A messages must be established (AIRTE). The options here are also ready list or input list.
8. The number of slots in the SLC routing table (NLRT), which will be described later, must be entered. This number must be equal to or greater than the number of table entries.

**Note:** Leave room for expansion.

9. Up to 50 entries may be coded as positional parameters to the SYNCLK macro for the SLC routing table. Each entry consists of 6 fields as follows:

- a. The symbolic link number (SKN) for this entry. The SKN is a relative number not an actual symbolic line number. For example, the first table entry should be coded as 1, the second as 2, and so on. The actual symbolic line number will be calculated by SIP based on the numbers of other lines on the system. The number of SKNs must not exceed the number of SLC lines entered in the keyword SLCAI in the SIP macro LINES.
- b. The high-level exit address associated with this pseudo-line number must be entered.
- c. The terminal circuit ID for this entry must be entered.
- d. The pseudo-line number for this entry must be defined. The pseudo-line number is also a relative entry processed in the same manner as the SKN. The number of pseudo-lines must not exceed the number defined by keyword PSLNS in the LINES macro.
- e. The interchange address on the pseudo-line must be entered.
- f. The data transfer code associated with the pseudo-line must be established. This code identifies the form in which data will be transferred as either padded SABRE, padded BAUDOT, International Standards Organization (Standard ISO 7 bit code) extended 7-bit, or extended padded SABRE. In addition, an indication must be made as to whether or not incoming messages should be passed to the input message editor package (UII). (See "UI/Reservation Package (PARS)" on page 149.)

Computer room agent set (CRAS) table (CRASTB macro): The CRAS table is established via this macro in order to allow communication between the TPF online system and system operators after an IPL has taken place.

1. The CRASTB macro provides for the definition of a prime CRAS device (the system console), a receive-only (RO) hardcopy terminal (system RO CRAS), and for both alternate consoles and alternate CRAS devices (alternate CRAS). The alternate consoles and alternative CRAS devices are used by the fallback routines if there is a hardware failure of one of the system consoles.
2. For LC users of the HPO feature, a prime and RO CRAS device must be defined for each processor in the complex.
3. In addition, input to the CRASTB macro also defines the physical type of console device to be supported by the system. Two generic types of devices are supported:
  - a. TPF supports the native system console, which is normally a 3270-like device (for example, a 3036, or a 3278 model 2A). This is referred to as native console support (NCS). In this case the RO CRAS device is a 3270 local printer and the alternate console devices would also be 3270 local terminals (both CRTs and printers).
  - b. TPF also supports the use of a 1052/3215 printer/keyboard which is attached to the system via an RPQed 7412 control unit. In this case the RO CRAS device may be a 1053 printer which is also attached via the RPQed 7412 CU. An alternate 1052/3215 console may also be specified which is used in case of hardware failure of the prime 1052/3215 console.
4. An additional function of the CRAS table is to allow the user to specify additional (99) alternate CRAS devices. These may either be 3270 local devices or terminals located out in the communications network. These are specified dynamically via the ZACRS command.

The terminals supported in addition to 3270 local CRTs and associated printers are:

- 3270 remote (SDLC) CRTs

- 3270 remote printers (3284 and 3286 devices)
  - 1977 typewriter terminals
  - 2915 and 4505 CRT terminals with associated printers
  - 1980-21 printers (for use with 2915 and 4505 CRTs).
5. CRAFTB macros are coded as follows:
    - Either NCS or 1052/3215 console support is selected via the NCONSL keyword parameter.
    - The prime CRAS (and alternate prime CRAS if 1052/3215s are used) subchannel addresses must be specified via the PRCRS and ALTPC keyword parameters.
    - The RO CRAS device is specified via the ROCRS keyword parameter.
      - a. If NCS is specified, the RO CRAS must be a 3270 local 3284 or 3286 printer device and only the subchannel address need be specified.
      - b. If a 1052/3215 is specified as prime CRAS, the RO CRAS device must be a hard-copy device (that is, a 1977, a 1980-21, or a 3270 local or remote 3284 or 3286 printer). The user may also code for ROCRS the same 1052/3215 typewriter prime CRAS device but its address must be coded as X'010002'. If any of the other devices are used, the address coded must be the device's line number, interchange address, and terminal address (LNIATA or LIT). For NEF and 3270/SDLC devices, the address is the pseudo LIT.
  6. For NCS and local 3270 CRAS devices, the user must specify the amount of time (MORE) that messages are held on the CRT screen before being cleared and the next message displayed.

**Note:** During online operation the initial CRAS configuration may be displayed or modified dynamically via the ZDCRS and ZACRS commands.

The number and type of non-SNA communication line/protocols (LINES macro) is used to define which and how many lines are attached to the host TPF system.

- The number specified is by type and should include spares for future expansion.
- The number specified for 3270 local (NLSLC) is the number of 3270 local terminals (not lines).

Additional non-SNA communications support (NETWK macro)—In addition to the previous SIP macros/keyword parameters, the NETWK macro contains additional keyword parameters which are of interest to the non-SNA communications designer.

1. The NETWK macro also contains the following keyword parameters:
  - a. The total number (N2703) of 3705 controllers plus the number of 3745 control units running PEP.
  - b. The maximum (M3270) size of an input/output message, including control characters, from/for any 3270 terminal.
  - c. The number (NUAT) of unique, including expansion, LNIATAs requiring an entry in the AAA/RCB initialization table (UAT). This includes pseudo LITs.
  - d. The time interval (WGTIME) for controlling the frequency of keypointing the WGTA table which contains the UAT entries allocated above.

Additional non-SNA communications support (CONFIG macro)—in addition to the normal communications macros specified above, the CONFIG macro contains several keyword parameters which are also of interest to the non-SNA communications designer. Note, MDBF users of the HPO facility should only code these parameters on the CONFIG macro which is associated with the BSS.

The CONFIG parameters are:

- a. Whether or not (MAPSP) support will be provided for the 3270 mapping support package. This is an application type support package which is further discussed in “TPF Mapping Support” on page 152. Support is provided for all versions of TPF’s 3270 support including 3270/SDLC, 3270 local.
- b. Whether or not (MSGSW) support will be provided for the message switching package (MESW). This is another application type support package which is further discussed in “Message Switching Packages” on page 152. It is also influenced by the CONFIG USER keyword (see below).
- c. Whether or not the TPF system is to be generated to run under the control of the VM product (VM=YES|NO) has communications impacts. The option should only be **YES** if the system being generated is a test system.
- d. Whether or not the system being generated is for either a domestic or a World Trade user (USER) governs the type of MESW support provided (see MSGSW above), plus unique support within several of the communications protocols. For MDBF users, one subsystem can be for domestic use and another for World Trade use.
- e. Whether or not the old PARS reservation function is included in the system is governed by the RES parameter (see “UI/Reservation Package (PARS)” on page 149). For MDBF users, one or more subsystems can have this unique application and others not have it.

SNA requirements—When defining an SNA communications network, including one with only NEF support, see the *TPF ACF/SNA Data Communications Reference*.

A TPF SNA network requires an additional network definition step, which is done outside of SIP. Refer to *TPF ACF/SNA Network Generation* for information on doing these steps.

In addition, there is also a requirement to do a separate generation of ACF/NCP. If 3600/4700 support is required, there are additional generation requirements to be able to load/dump these communication devices.

Message router support (MSGRT and MSGRTA macros)—Communications support requires the use of the TPF message router and log processor packages. The SIP requirements for this support are discussed later in “Message Router Support” on page 142 and in “Log Processor” on page 145.

## Communication Keypoint Macros (Non-SNA)

The non-SNA communications network is defined for the system by the following set of keypoints:

- System communication keypoints (SCK)  
The SCKs contain the status of each non-SNA communication line attached to the system. They are used at restart time to build the following main storage resident communication tables:
  - Line status table (LSTB)
  - Terminal interchange status table (TITB)
  - Symbolic line status table (SLST)
- Communication control unit (CCU) keypoint status record (PKST)  
The PKSTs contain the status of each 3705 EP or 3745 PEP on the system.

The SCK and PKST generation package (see “Non-SNA Communications” on page 335 ) provides a set of macros and an offline process which can be used to define the user’s non-SNA network configuration and produce STC input to create a communications pilot tape which contains an SCK for each non-SNA line and a PKST for each non-SNA communications control unit. Note, communication keypoint records are not required for a network supported only by the Network Extension Facility (NEF) and SNA.

Refer to Table 21 on page 134 and Table 22 on page 136 to assist in setting up SIP and SCK/PKST inputs. Information entered in SIP macros is passed to the SCK/PKST macros via the SYGLBK and SYSETK members which are created during SIP Stage II execution. This insures basic integrity between the two processes, SIP and SCK generation. SIP Stage II must be successfully completed prior to SCK/PKST generation.

- SCK generation macro statements are used as input to the MVS assembler to generate STC input for SCK phase II. A set of two or four macros depending on the line type are required to generate the SCK record for each line.
- The following macro set is required to generate an SCK for a line:
  - SKLNG** Defines the symbolic line number and paths.
  - SENDG** Generates the SCK/STC input.
- PKSTG generation macro statements are used as input to the MVS assembler to generate STC input for SCK phase II. One PKSTG macro call is required to generate the PKST record for each control unit.

The SCK macro sets must be coded in symbolic line number (SLN) sequence and the line numbers must start at two. SLN number one is reserved for use of prime and RO CRAS. The SCK input for these unique entries is described in “Communication Pilot Record Generation” on page 137. This is to ensure that all the SCKs are initialized. If the macros are not coded in line number sequence, the user must ensure that an SCK has been created for every line in the system. PKSTG macro calls must follow the last SCK macro set. All control unit numbers must be in sequence order and the control unit number must start at zero. If PKSTG macros are not coded in control unit number sequence, the user must ensure that a PKST has been created for every control unit in the system. A SENDG END macro statement must follow the last PKSTG macro call; this will cause final validation checks.

During phase I of the SCK process the user coded SCK/PKST macro statements are analyzed for errors. Some errors will prevent SCK/PKST record generation from continuing but in most cases errors will be flagged and processing will continue, using default values wherever possible, to produce an assembly listing and STC input for phase II. If errors occur in phase I, they must be resolved before proceeding to phase II.

Phase II is a straightforward STC execution of the output of phase I. Before executing STC, the unit test disk, DISK01, should be loaded with an SDMF and a DRIL file containing at least the following macros (see note 3):

- SCKDS** Data record description for SCK.
- CPTIC** Data record description for PKST.

Output from phase I can then be used as input (CARDIN; see note 1) to STC to produce a communications pilot tape (TAPE1; see note 2). This tape is then used as input for the online system loader to initialize the communications keypoint records on file.

**Notes:**

1. CARDIN is the DDNAME given to the DCB describing the input file of STC executing under MVS control.
2. TAPE1 is the DDNAME given to the DCB defining the output tape file from STC.
3. Loading of the SDMF and DRIL disk can be accomplished at the time the SCK pilot is created by coding the parameter 'LOAD=YES' of the SENDG macro. The user must also insure that the DD statement for the SDMF and DRIL tape is included in the STC execution JCL.

Two worksheets are provided to aid in the definition of the non-SNA communication network. The entries for the transmission control unit worksheet (Table 21) and the lines definition worksheet (Table 22) are described below. An individual entry is made for each terminal control unit and each line in the non-SNA network. No entries need be made for the NEF supported network.

On the transmission control unit (TCU) definition worksheet (Table 21) the user must describe the physical layout and characteristics of all TCUs on the system being generated. SIP and PKST macros both require inputs. Information entered in SIP macros is passed to the SCK/PKST generation package which insures basic integrity between the two system generation processes. Outlined below is a brief description of each column in the figure and the related SIP and PKST entries required:

1. Indicate the symbolic control unit number in column one.
  - PKST Input = PKSTG (CUNO)
2. Enter the control unit type in column two.
3. Indicate the initial status of the TCU as either online or offline in column three.
  - PKST Input = PKSTG (CUSTAT)
4. For 3705 EPs and 3745 PEPs, indicate the native subchannel address of each TCU along with their associated channel adapter type in column four.
  - SIP Input = NETWK (SUBCH)
5. Indicate the range of subchannel addresses associated with each TCU in columns five and six.

Finally, the total number of 3705 EPs and 3745 PEPs defined in the TCU worksheet is required as input to SIP.

- SIP Input = NETWK (N2703)

The first entry has been entered as an example for the user.

*Table 21. Non-SNA Transmission Control Unit (TCU) Worksheet*

Symbolic CU Number	CU Type	TCU Status Online=ON Offline=OFF	3705 Native Subchannel Address *	Lowest Subchannel Address	Highest Subchannel Address
				60	6F
02	3705	ON	1A-1		



Table 21. Non-SNA Transmission Control Unit (TCU) Worksheet (continued)

Symbolic CU Number	CU Type	TCU Status Online=ON Offline=OFF	3705 Native Subchannel Address *	Lowest Subchannel Address	Highest Subchannel Address
<b>Note:</b> Total number of 3705 EPs and 3745 PEPs = _____ * Column 4 is not needed for 3745/PEP.					

All non-SNA lines on the system being generated must be declared in SIP and defined with SCK macros. As with TCUs, data received in SIP is passed to the SCK/PKST generation package through SYGLBK/SYSETK. Lines declared in SIP must be defined in SCK/PKST so that an SCK data record is created for each line.

The worksheet in Table 22 on page 136 is provided to serve as a guide and format for designing line definitions. All columns correlate to the narrative below in the same manner as the TCU worksheet above. Inputs to the worksheet are as follows:

1. Indicate the line type (SLC, LC, or BSC) for each line in column 1.  
SIP Input = LINES (line type mnemonic)
2. Indicate the symbolic line number in hex for each line in column 2. The SLNs are created in direct relation to the line type and number of lines in the line status table (LSTB). At the completion of the SIP run, the SIP report lists the line types and their associated SLNs under the heading labeled "Lines."  
SCK Input = SKLNG (LNSY)
3. Indicate the paths (from 1–4) by which a line may be connected to the TPF host in columns 3–7. The maximum number of paths described for any line must not be greater than the value entered in the SIP NPATH parameter of the NETWK macro.  
SCK Input = SKLNG (PATHx)  
Each path requires the following information:
  - The physical line number of the path entered in column 3.
  - The symbolic control unit number to which the line is attached, entered in column 4.
  - The control unit type (3705 EP, 3745 PEP, or 3270 local) in column 5.
  - Enter in column 6 a 3-character code as follows:  
1st character = (L) 3270 local line, 3745 PEP, or 3705 EP  
2nd character = (A) automatic switching status  
(M) manual switching status

3rd character = (P) primary path  
(A) alternator path

Example SCK Input: PATH=(,LMA)

4. Enter in column 7 the count of the number of BSC messages that will be sent before line turn around occurs and messages will be received.

SCK Input = SKLNG (SL)

5. Indicate for BSC lines the line type; either multipoint (MP) or point-to-point (PP) and whether or not the line has blocking (Y or N) in column 8.

SCK Input = SKLNG (BTYPE)

6. Indicate for BSC lines the type of code used as follows in column 9.

E/U = EBCDIC or UASCII code

Y/N = Data transparency, yes or no

Y/N = RCPLs required, yes or no

SCK Input = SKLNG (BCODE)

7. Indicate for BSC multipoint lines the time-out value (0–255) in column 10.

SCK Input = SKLNG (BSTOV)

For pseudo lines, only 1 path is required. They should be defined in the SKLNG macro as PATH1 = (ppp,nn.LMP) where ppp equals the physical line number, nn equals the actual symbolic control unit number through which the line passes, and LMP indicates the following:

L = 3705EP

M = manually switchable

P = primary path.

For 3270 local lines, only 1 path is allowed. It should be defined in the SKLNG macro as PATH1 = (ppp,,LMP) where ppp equals the physical line number and LMP = local 3270, manually switchable and primary (only) path.

Finally, all non-SNA terminals or terminal control units must be defined with SCK generation macros. The following are the terminal parameters required for BSC:

- BSC
  - SCK Input = BSTxx (POLCH)
  - SCK Input = BSTxx (SELCH)
  - SCK Input = BSTxx (STANM).

Table 22. Non-SNA Communication Lines Worksheet

1	2	3	4	5	6	7	8	9	10	Description
Line Type	SLN	Line Paths (1–4)					BSC			
		PLN	SCU	CUT	LNS	SL	TYP	COD	TIM	
–	00									RO CRAS
–	01									Prime CRAS
BSC	43	032	03	3705	LMP	101	PP-B	ETY	50	
LC	50	094	06	Local	LMP					



Table 22. Non-SNA Communication Lines Worksheet (continued)

1	2	3	4	5	6	7	8	9	10	Description
Line Type	SLN	Line Paths (1–4)					BSC			
		PLN	SCU	CUT	LNS	SL	TYP	COD	TIM	
Legend:										
SLN	Symbolic line number									
PLN	Physical line number									
SCU	Symbolic control unit number									
CUT	Control unit type (3705 EP, 3745 PEP, or 3270 local)									
LNS	Line status									
SL	Line turnaround count									
TYP	BSC line type									
COD	BSC line code									
TIM	BSC multipoint timeout value.									

## Communication Pilot Record Generation

The system test compiler (STC) is a multipurpose tape file generator that is used to create pilot tapes. See *TPF Program Development Support Reference* for additional information about this package. STC is used to create a pilot tape containing the following non-SNA communication records in addition to its use in SCK generation phase II, described previously.

**AAA/RCB initialization table (UAT):** This table contains an item for every terminal (including NEF-supported terminals) and every binary synchronous line in the system. Each item contains the necessary data to initialize the agent assembly area (AAA) and the routing control block (RCB) for the terminal or line it references. The UAT table is used to initialize the AAA and RCB records. For MDBF users, a set of UAT records must be built for each subsystem. The set loaded to the basic subsystem includes all terminals. The set that must be loaded to a subsystem is a subset of the basic subsystem's UAT records. This subset will contain only entries for terminals that will have access to that subsystem. This subset will be used for the record initialization described previously and for the initialization of the subsystem ordinal table records (SSOR). See "Operator/Terminal Control" on page 146 for an explanation of SSOR use.

The UATs are also used at restart time to build the WGTA table where there is one entry for each UAT. The SIP input for the NUAT keyword of the NETWK macro specifies the total number, including expansion, of UAT entries that are now being created. The WGTIME keyword is also associated with the WGTA table created from the UATs.

If an automation gateway is connected to a processor for multiple console support, you must specify additional UATs for that processor. Besides the Prime CRAS and RO CRAS UAT entries, you must generate UAT entries for each possible LNIATA made up of LN/IA 0100 and the TA in the multiple console support header received from the automation gateway; that is, UATs for LNIATAs 010003 through 0100 $mm$ , where  $mm$  is the maximum terminal address specified in your IBM Extended Operations Console Facility/2 (EOCF/2) configuration. (For additional information about the maximum terminal address, see the *IBM Extended Operations Console Facility/2 System Administrator's Guide*.) Unlike those for the prime CRAS and RO

CRAS, the additional UATs should be generated as **not** permanently logged to ensure that the other consoles can log on or off from applications other than SMP.

XLMT assembly area (XLMA)—These areas are 381-byte records located in fixed locations on file and used by the LMT package. (See “Long Message Transmitter Package” on page 151.) An XLMA record is reserved for each hardcopy terminal (for example, IBM 1052, 3215, 1980-21, 1980-24, 1977, 3284, or 3286) whose address is referred to by symbolic line number, interchange address, and terminal address (LNIATA). An automation gateway is an exception; LNIATAs 010003 through 0100 $mm$  do not require an XLMA record, where  $mm$  is the maximum terminal address specified in your IBM Extended Operations Console Facility/2 configuration.

These XLMA records are used to queue and control transmission of messages to these terminals.

On a WTC system using IPARS message switching, a record must be assigned for each CRT device defined as a message switching input terminal.

Entries must be made in the XLMT assembly area for all hardcopy terminals supported by the network extension facility (NEF). The NEF pseudo LN/IA, along with the real TA, should be used.

Line number/interchange address/terminal address index (XLAD): The XLAD table is maintained in the unprotected application global area and is used to convert the high-speed line number, interchange address, and terminal address (LNIATA) into an ordinal number. The long message transmitter program (LMT) uses this number to access the XLMT assembly area (XLMA) assigned to each:

1. LNIATA hardcopy terminal.
2. CRT devices defined as message switching input terminals. This applies only to World Trade systems using the IPARS messages switching package.
3. NEF-supported hardcopy terminals require an entry using the NEF pseudo LN/IA and the real TA.

Agent assembly area—Each AAA is a record permanently assigned to a specific terminal in the system. Pseudo AAAs not associated with a terminal are also temporarily created for internal use (for example, TTY or file record processing). AAAs are required for the PARS (RES0) and IPARS applications. If an automation gateway is used and the local area network (LAN) terminals are logging into airline reservation applications, AAAs must be allocated for all terminals with LNIATAs in the range of 010003–0100 $mm$ , where  $mm$  is the maximum terminal address specified in your IBM Extended Operations Console Facility/2 configuration. When MDBF is installed, AAA records only have to be allocated in the subsystems that have an airline reservation type of application.

If an automation gateway is used, all the possible terminal appearances on the LAN require an RCB; that is, LNIATA 010000 and LNIATA in the 010002 to range 0100 $mm$ , where  $mm$  is the maximum terminal address specified in your IBM EOCF/2 configuration. (For additional information on the maximum terminal address, see the *IBM Extended Operations Console Facility/2 System Administrator's Guide*.)

System communication keypoints: These keypoints are normally generated using the SCK/PKST generation package (see “Communication Keypoint Macros (Non-SNA)” on page 132). However, systems supporting only SDLC communication

lines still require 2 SCK records for the prime and RO CRAS terminals that must be generated via STC. The following is the required STC input:

```
*
*   SCK RECORDS FOR PRIME AND RO CRAS
*
SCKDS   GSTAR 1.
BSTA06  ENT   (#PKSCK)0.
SCKBID  ENT   SK.
SCKIND  ENT   X'80'.
SCKDIS  ENT   X'003E'.
        GEND
SCKDS   GSTAR 1.
BSTA06  ENT   (#PTSCK)1.
SCKBID  ENT   SK.
SCKIND  ENT   X'80'.
SCKDIS  ENT   X'003E'.
        GEND
```

The following are the fields that must be initialized for the previous record types:

Record Macro Name	Mandatory Fields to be Initialized	Title And Notes
XS0AA	XS0AVA XS0BCH XS0CAP XS0CCT XS0COP XS0FCH XS0LNR XS0LOP XS0MIA XS0MSG XS0MTL XS0OLN XS0OME XS0OTC XS0OTL XS0OTM XS0RMC XS0RSC XS0SOP XS0SP1 XS0SP2 XS0SSP XS0TCC XS0TQC XS0TSI XS0TTA XS0TTP	XLMT Assembly Area
XX1ON	XX1ELN XX1IN1 XX1LNR XX1TLN XX1TON	LNIATA Index
UA1UA	UALIND UA1ADD UA1APN UA1CID UA1CIT UA1ISP UA1TTP UA1TYS UA1XRF	AAA/RCB Initialization Table—One item for each high-speed terminal including NEF supported terminals.
CI0CO	CI0IAS CI0LNB CI0MOD CI0TAS CI0TTP CI0TYS	Routing Control Block—One record per terminal.
WA0AA	WA0CIT WA0GFC WA0HFC WA0IAS WA0IOT WA0ITC WA0LNB WA0LST WA0NAB WA0NMC WA0RFC WA0RKC WA0SIN WA0SIO WA0SIS WA0TAS WA0TKC WA0TLC WA0TNC WA0TYS WA0TY1	Agent Assembly Area—One completely initialized AAA for a prime and RO CRAS set must be program (UAA1). Required only if created by the AAA initialization RES=YES.

## SNA Table Generation

SNA table generation, or SNA network definition, is performed by the offline SNA table generation (OSTG) program and by the SNAKEY macro in the SNA communications keypoint (CTK2). See *TPF ACF/SNA Network Generation* for detailed specifications about performing these two tasks.

SIP provides input to the SNA table generation program in the form of a data set name that contains the application name table (ANT) definitions, and is derived from the MSGRTA macros of SIP as coded by the user. (See “Message Router Support” on page 142.) The SIP GENSIP macro provides the ANTPDS parameter, which is used to specify the actual name of the data set that will contain the ANT entries for the SNA table generation program.

Online, SNA restart initializes the main storage resident resource vector table (RVT) from the resource resolution table (RRT). The main storage resident copy of the subarea address table (SAT) is built from the SAT file created offline by OSTG.

## 3600/4700 SNA Generation Requirements

When you use either the 3600 or 4700 Finance Communication systems, both PUs need to be loaded and dumped, and this must be done by another system. The component that does this function is known as subsystem support services (SSS). You must generate the system so that these devices are owned by the TPF-required VTAM CMC and, as a result, all loading or dumping is taken care of by that system. The encrypt/decrypt module desired should be added to the TPF-released object library for any system supporting 3614s.

The SIP CONFIG macro provides the CIPHER parameter, which is used to specify which of the following encryption techniques are desired (both, one, or the other, or none):

- National Bureau of Standards proposed U.S. Federal Information on Processing Data Encryption Standard (DES)
- Original IBM 3614 encryption technique (module BQKCIPH), which is referred to by SIP as the alternate encryption technique (AET).

Additional information regarding SSS can be found in *3600 Finance Communication System 3614 Programmer's Guide and Reference*.

Application programs that use TPF SNA support as their communications medium must conform to those SNA disciplines implemented in TPF unless it is the NEF support. The *TPF ACF/SNA Data Communications Reference* describes host application programming and SNA message protocols that may be implemented. The *IBM 3600 Finance Communication System Instructions and Macro Reference*, and the *IBM 3600 Finance Communication System Programmer's Guide and Component Descriptions*, should be helpful to the application developer. Resources defined in a particular network mode must be examined carefully for dependencies that must be defined in other interfacing nodes. For example, input buffers must be defined so that they are compatible with ACF/NCP-defined output message lengths.

---

## Application and System Support

Many of the areas addressed in this section of this document could be classified, in some respects, as application support. However, there are several areas that are designed specifically as interfaces between the hardware and system programs, and between the application programs and system programs, which will influence the way application programs are designed.

Many of these areas, which are often referred to as *packages*, must either be included or excluded from the generation of the TPF system via the use of keyword parameters that are associated with various SIP macros. Several require that specific fixed file records be allocated for their use. In some cases, there are requirements to initialize records which are then loaded to the system via pilot tapes (see “Communication Pilot Record Generation” on page 137). There are even cases where the user may be required to physically update the source code of system programs to achieve specific results, and other cases where defined user exits have been provided.

In summary, the overall purpose of this section of the document is to introduce these various packages, to explain their basic purpose, and to explain the generation requirements for them.

---

## Application Program Interface

The interface between TPF and the user application programs is defined by TPF's message router support (see “Message Router Support” on page 142). This is true for all applications except those associated with specific 3705 EP line protocols (for example, the message switching package).

When an operator/terminal, or even another application program, is logged to an application and a message is received from that terminal or logical unit, control is passed to that application, or more precisely to a message editor program defined for that application, based on an entry in the Routing Control Application Table (RCAT). The RCAT entry defines the application as valid and active, and specifies the editor program to be entered for initial processing. The RCAT for MDBF users of the HPO feature, also specifies which subsystem and subsystem user that the application resides in. The RCAT is one of the components of the message router. It is initialized by SIP with an entry for every application the user intends to activate. Applications may be activated or deactivated via a command. Refer to “Message Router Support” on page 142 for additional details about message router components and its generation requirements.

When the application input message editor is given control the following standard interface is presented:

- The first 12 bytes (16 bytes if extended format is specified) of the entry control block (ECB) work area (EBW000-EBW011) contain the routing control parameter list (RCPL).
- Core block reference word zero (CE1CR0) of the ECB and general register 1 contain the address of the input message in main storage.
- Core block reference word three (CE1CR3) may contain the address of the terminal's routing control block (RCB) in main storage if the input message originated at an ALC terminal and the application requires a terminal control record. The associated RCB fixed file record is in HOLD status with the record ID, ordinal number, and file address located in ECB field CE1FA3.

- If the message originated at a non-SNA terminal, then field CE1CPD of the ECB contains the ID of the TPF host in direct control of the terminal. If the source of input is another application then field CE1CPD contains the ID of its own TPF host.
- If the message originated at a non-SNA terminal, then field EBROUT of the ECB contains the terminal ID (that is, LNIATA) of the activating terminal.  
If it is a SNA terminal, not logged to an application using an LNIATA rather than a resource identifier (RID), then field EBROUT contains the RID associated with the SNA logical unit (LU).
- If the message originated from an SNA terminal, not logged to an application using an LNIATA rather than an RID, then the field EBROUT contains the RID associated with the SNA terminal (Logical Unit - LU).
- For users of MDBF of the HPO feature, various fields within the ECB indicate which subsystem and subsystem user that the application is associated with. For non-MDBF users, these fields are set for the basic subsystem.

The concept of the message router interface is used not only for relating input messages from the communications network to application programs but it is also used to pass messages between one application program and another. These applications may be located within this TPF host or, with the use of ACF support, in either another TPF/ACF host, which could be within a loosely coupled complex, or any other defined host system. For MDBF users, this same technique should be used to pass messages between subsystems and subsystem users.

For a more detailed discussion of TPF's application support, refer to the following TPF documents:

- *TPF Concepts and Structures*
- *TPF ACF/SNA Data Communications Reference*
- *TPF Application Programming*.

---

## Message Router Support

As a result of users coding the SIP MSGRT and MSGRTA macros, which define the message router support, program segments are produced by SIP. These program segments are collectively referred to as the RCAT Initialization Table (RCIT).

The RCIT defines:

- The application program names that will be available in the network.
- The host in which an application will be resident.

**Note:** The applications residing in a loosely coupled complex are defined to be resident in each processor of the complex.

- The attributes/restrictions of the various applications.
- The subsystem user in which an application will reside (an index to the SSUT for MDBF users).

The non-propagable bit in the RCIT is set to 1 by the SKRCIT internal system generation macro.

The initialization of the previously mentioned router records, except for SSUT which is created at restart time for MDBF users by the IPL program, is performed at generation time by SIP from information provided by the user.



If all the applications defined via SIP reside in a single non-loosely coupled TPF processor then these macros alone will generate the proper level of message router support. If, however, the generated TPF system requires loosely coupled support, **or** any of the applications reside in a different processor complex, then the ACF support is also required for SNA support, and OSTG must be run to define the required Functional Management Message Router (FMMR) entry and path for interprocessor communications. This includes loosely coupled systems that have no other need for SNA support, since the FMMR entry in the SNA Resource Vector Table (RVT) is used to control the interprocessor communications between the processors within the loosely coupled complex. Refer to "Message Router Support in an ACF Network" for additional information.

## Message Router Application Attributes

The MSGRTA macro, in addition to defining an application and its residency in an host, also allows the user to define unique attributes for the application itself. A major difference between applications is the line protocol used. SNA, or new applications, can use the SNA resource identifier (RID) to identify a SNA logical unit (LU), or can use a LNIATA to identify terminals. Non-SNA, or old applications, use a terminal address such as a LNIATA to identify terminals. The message router does provide the ability for 3270/SDLC terminals to enter old applications via a pseudo-LNIATA (PLIT). All of the attribute keywords are not mandatory with defaults provided. The defaults are for SNA application programs.

The keywords/attributes are:

1. RES=YES defines reservation type of application such as PARS (see "UI/Reservation Package (PARS)" on page 149).
2. SIGN=YES specifies that there is a user written/controlled sign-on procedure (see "Log Processor" on page 145). When a terminal operator logs into this application, a response will be sent requiring sign-on.
3. SIMFM=YES specifies that 3270 terminals may be used with an application that only has been coded to accept input from ALC terminals. The input is reformatted by the system from 3270 to 4505 format before being presented to the application. The system also reformats output messages from this application to 3270 format.
4. ASNA and TERMRCDD specify the type of terminal record and whether or not it is to be actually passed to the application program.
5. MRECV specifies for SNA applications whether or not SNA system message recovery is to be performed for the application. The application is associated with message recovery support or specified on SNAKEY in CTK2.
6. RCPL specifies the size of the routing control parameter list. SNA logical units may use the extended RCPL.
7. ALTBUFF=YES specifies the use of the alternate buffer size by the specified application for 3270/SDLC logical units.

## Message Router Support in an ACF Network

In an ACF network, all messages destined for an application resident in another processor will be forwarded to that processor via the FMMR. The FMMR is defined via the offline OSTG program (see "SNA Table Generation" on page 139).

In a loosely coupled complex, messages flow on the IPC path.

NEF users of the message router are treated and generated like ALC users. The terminals supported by NEF (ALC terminals) will use an LNIATA when they log into applications.

## Message Router Support for the High Performance Option Feature

Loosely coupled users of the HPO feature must generate the message router, taking into consideration that they are using ACF support. All local applications are defined as existing in all processors in the LC complex. The system applications (for example, SMPx) exist uniquely in each processor of the complex and are identified by their appended TPF processor ID. To implement this, the APROC keyword of the MSGRTA macro is coded with the \* character. The actual TPF processor ID comes from the IDs specified on the SYSID keyword of the CONFIG macro.

MDBF users must indicate the subsystem and subsystem user with which the application being defined by the MSGRTA macro is to be associated. This is done via the USER keyword parameter of the MSGRTA macro, and it must match the name associated with one of the coded SSDEF macros.

In addition, at least 1 application in each subsystem must be defined as an SMP-like application via the SMP= keyword (see “Communications Source Message Router and System Message Processor” on page 148). This is the application that will receive the commands for that subsystem. For the BSS, the message router automatically provides the SMPx application.

## Message Router Implementation Guidelines

The following items should be considered:

- The message router uses a unique format on the communications lines for communicating with other systems (see data macro PP0SG). When the other system is a non-TPF system, it is the user's responsibility to provide the necessary programs in the non-TPF system to properly process these messages.
- The TPF system console (prime CRAS) and receive-only consoles (RO CRAS) must be attached to the TPF processor in which they are so designated.
- In addition to the RCIT entries generated for the user-specified applications (MSGRTA macros), SIP automatically generates RCIT entries for the following application names:

**LOGI** Log processor

**SMPx** System message processor

**CLGx** Sign-on interface

where: x represents the processor ID of the TPF system.

- Program segment COBD assumes that the pool records are allocated for the following IDs:

**LL** Large, long-term pools.

**SL** Small, long-term pools.

**LS** Large, short-term pools.

**SS** Small, short-term pools.

Review IBMPAL to ensure that these IDs are defined properly for each TPF subsystem.



- If routing control block (RCB) records are in the system, they must be initialized via the initialized RCB command (ZRCBI) before running recoup. This is to avoid dual usage or dual release of pool records or both.

---

## Operator, Terminal, and LU Identification and Control

The purpose of this function is to identify and control operators, terminals, or SNA logical units. This may be accomplished at two levels. At the gross level, the terminal is simply assigned to an application and the operator authorized to input data to it. This level is handled by the log processor. Some applications will require more precise control for which a group of records and support programming is available to facilitate user tailoring. (Application sign-on is level 2.)

### Log Processor

The log processor component of the message router package performs the connection or disconnection of a terminal/logical unit to or from an application. This connection establishes a data path between the terminal and the application so that, independent of the physical location of the application in the network, all the data entered at the respective terminal/logical unit is delivered to the *logged* application for the duration of the connection.

Normally the connection is established by a *login* message (or *logon* for ACF/SNA), and ended by a *logoff* message. However, SNA logical units, except for NEF terminals, may be permanently logged to a specific application. Non-SNA terminals may be associated with 1 or more applications through the use of a log authorization table. This indication is recorded in the AAA/RCB initialization table (UAT), which is generated as part of the pilot tape (see "Communication Pilot Record Generation" on page 137).

All valid application names must be recorded in the routing control initialization table (RCIT), which also contains for each application: status information and the maximum number of logs allowed. Before a terminal/logical unit can log on, the application must be made active via a command. Once the log is successfully completed, the index to the application name table (ANT) is stored in the WGTA entry for the terminal. For SNA support, the index to the ANT is stored in the resource vector table (RVT). The application name table is a single main storage resident record containing up to 255 application names along with the respective addresses of the RCAT application names and the addresses of the RCAT entries.

Two levels of security may be implemented via the log processor. The security level will be indicated in the RCAT entry for each application. The two levels are as follows:

1. At the first level, the application may be used only by authorized terminals. This is controlled by authorization indicators in the WGTA, for all terminals. There are three types of terminal authorization security:
  - a. Permanently logged, as described previously.
  - b. A terminal may be allowed to log on to all applications.
  - c. A terminal may be allowed to log on to a specified list of applications. This list is contained in the terminal application authorization table (TAPP). There is one set of TAPP records for the system. Users must create the TAPP records either offline, on a pilot tape using the system test compiler, or online using the ZAUTH command.
2. The second level of security, one designed uniquely for and by the application itself, may be required. This may be needed for accounting purposes, or for

applications performing multiple functions, each of which require unique security codes, or for a variety of reasons known only to the application. This is not a system function but an application one; it must be designed by the user, and it is termed the *application sign-on/sign-out procedure*.

### Sign-In/Sign-Out Procedures

When the RCAT entry for the application indicates sign in or sign out is required, an entry point is provided so that the application can notify the log processor of successful sign in. Once the log processor is so notified it no longer monitors messages from that terminal, but they are passed directly to the application until such time as the operator signs out and the log processor is so notified. The interface to the log processor is in the form of an application-to-application data transmission where:

- The origin of the data is the application itself.
- The destination of the data is application CLGx (where x is the processor ID of the TPF host in direct control of the terminal). If the application sending the notification resides in a non-TPF host, the processor ID would have been passed to it as part of the origin of the data in the processor-to-processor message format header (data macro PP0SG). If the application sending the notification resides in a TPF system, the processor ID would have been part of the origin in the RCPL.
- The text of the data is the address (LNIATA CPUID) of the originating terminal.

For more details about the log processor, see *TPF Data Communications Services Reference*. For the SNA log processor see the *TPF ACF/SNA Data Communications Reference*.

## Operator/Terminal Control

Over and above logging or sign-on/sign-out, TPF provides facilities for more precise terminal or operator control related to the input/output message stream and application work areas. These facilities are different in the SNA and non-SNA support packages.

Non-SNA support (including NEF):

1. For ALC terminals, an agent assembly area (AAA) (or for 3270 local terminals a routing control block RCB) is permanently assigned (in the fixed file area) to each terminal attached to these lines. In addition, an RCB is assigned to each symbolic line (SLN) used in a BSC link for maintaining common data relative to multiple transmissions such as message header information and data chaining fields. No RCBs are assigned to SLC links, BSC stations, or SNA logical units (SDLC terminals). The TPF host processor ID renders the RCBs unique in the communications network.
2. Therefore, each RCB is identified by an address (LNIATA or SLN) and a processor ID. 3270 SNA logical units (SDLC terminals) with pseudo LNIATAs have RCBs assigned in the same manner as real LNIATAs. In order to retrieve a RCB, the communication source program invokes the read AAA/RCB program, which uses the WGTA to retrieve the address of the AAA or RCB.
3. For MDBF users, each subsystem will have its own unique set of AAA/RCB records. It is probable that each subsystem will also have its own unique set of ordinal numbers for these records. In other words, a terminal that has access to multiple subsystems will most likely have a different record ordinal number in each subsystem. The subsystem ordinal table (SSOR) will be used by WGR to cross-reference a terminal to a subsystem ordinal number.

4. If the source of data is a BSC link, the RCB is retrieved only if a multiple block transmission is received and, in that case, the RCB is used for message assembly and is not passed to the application program.
5. The RCB contains a system area and an application work area. The system area contains a message assembly area and control fields for proper routing of messages to and from that terminal. For BSC the RCB serves solely as a message assembly area.
6. Each RCB is initialized, when its terminal or link becomes active in the system, by the RCB initialization program (RCBI) from information contained in the RCB/AAA initialization table (UAT). Users must create the UAT using the system test compiler.
7. The agent assembly area (AAA) uses the same concept, organization, and accessing method as the RCB. The same UAT and WGTA tables are used for the AAA and for the RCB, simply with a different record type. Non-LNIATA systems do not require AAAs or RCBs.

#### SNA support:

1. For messages arriving via SDLC lines, logical unit control, logging information, and message assembly are handled by system records not used by the application; the resource vector table (RVT) and the node control block (NCB). These records must not be used by application programs and are not passed to the application input message editor program when the application is activated.
2. An application may require a record for work space and for control of message input/output related to the logical unit. When the logical unit consists of multiple work stations, this will usually be essential. The scratch pad area (SPA) is designed for this purpose. There is one for each SNA resource in the network. It may be either a 381-or a 1055-byte record and is undefined except for a standard header and 13 bytes of control data defining the node. All SPAs are initialized via command (ZNSPA) by the SPA initializer program that (1) writes the header (2) copies the 13-byte data item from the resource resolution table (RRT) and (3) zeros the remainder of the block. User responsibility is to allocate the required space on the fixed file storage for the record size that fits the user requirements. The definition of the body of the record and any required initialization is also a user responsibility and normally should take place in the application input message editor.

In summary, user requirements for implementing operator or terminal control records are:

- Allocate the required number of fixed file records via SIP for the chosen configuration, application, and design for the following record types, if used:
  - Routing Control Block
  - Agent Assembly Area
  - AAA Initialization Table (UAT)
  - Scratchpad Area
  - Operator Authorization Table
  - Node Control Block
  - Terminal Application Authorization Table
  - Subsystem Ordinal Record.
- Create the pilot records, using the system test compiler, for initializing the following records, if used:
  - AAA Initialization Table (UAT)
  - Operator Authorization Table

## Communications Source Message Router and System Message Processor

The communications source package receives input messages from the various line discipline programs and from application programs for entry into the system. As appropriate it provides for:

- Message translation and character deletion
- Message assembly and integrity
- The origin/destination connection, including user determination for MDBF
- Activation of the message router package or direct activation of an application if it is locally (this TPF host) resident.

If the user needs to modify the routing of input messages, there is a capability to write a user-specified exit routine to the communications source package which gets control upon each input message processed by the package. Use of this option is specified by using either the COMEXIT=YES keyword of the MSGRT macro or by using the ZSYSG command. In addition, the user must code the required logic into message router segment COBC. This segment is released with only a BACKC macro in it.

The router facility can be used with TPF tightly coupled (TC) support. TC support allows applications to run on more than a single CPU on processors with multiple CPUs. COBC, in addition to changing the destination application or the content of the input message, can direct the input message to a particular CPU (also known as I-stream) in the processor. Specific details for directing a message to an I-stream are given in the COBC commentary. When the default for the I-stream (0) is used, the I-stream scheduler selects an I-stream to balance the load between active I-streams.

For example, a user could write an exit routine to change the application input message editor program that is to get control, based on a unique primary action code in the input message itself. Therefore, a terminal logged to a particular application would not have to log off of that application and log on to another one simply to enter a single input message if the user exit routine could detect this condition based on the unique text in the input message. This facility is especially useful to MDBF users where the applications may be in different subsystems.

The system message processor (SMP) is a unique system-generated message router application that provides input and output processing of system control commands. It receives such input messages and activates the appropriate handling program (CVAA). All responses to commands and all unsolicited system messages are also transmitted by SMP.

The system operator console (prime CRAS) is permanently logged to the SMP application of the TPF host to which it is connected. All terminals logged to SMP, including the prime CRAS, have the capability to enter input messages to any non-SNA application by specifying the application name at the start of the message. This function is known as *message prefixing*.

In addition, each CRAS terminal can be designated via the ZACRS command to handle input or output for particular system functions. This is done via tables in the

SMP command editor that specify a *function code* as well as the name of the segment that will service the message. The released SMP supports 7 unique *functions*:

- Prime CRAS (of the BSS in an MDBF system)
- RO CRAS (of the BSS in an MDBF system)
- A separate tape functional console
- A separate DASD support functional console
- A separate communications support functional console
- An audit trail console
- Real-time database services (RDBS), which is used by the TPF Application Requester (TPFAR) feature and the recoup utility.

These designations are done via assembler EQUATE statements in the RTCEQ macro and then entered as a 2-byte field for each unique command. The user may add additional functional areas by making use of the spare RTCDFCSx equates in the RTCEQ macro and updating the SMP command editor table (CVAB) to designate which commands will be sent to the new functional console. Specific information on how to implement additional functions as part of functional console support (FCS) may be found in *TPF Data Communications Services Reference*.

HPO feature users of either the loosely coupled facility (LC) or the multiple database function (MDBF) have unique requirements for SMP as follows:

1. Each processor in the LC complex contains the exact same message router applications except for the system-generated applications, including SMP. In each processor there is an SMP application named SMPx, where x is the TPF processor ID that has been specified on the SYSID keyword of the CONFIG macro.
2. MDBF users, in addition to the SMPx application that is automatically generated for the basic subsystem, must include an SMP application in each subsystem. This is done via a combination of the USER and SMP keywords on the MSGRTA macro. This is necessary to provide each subsystem with an input command capability. This also means that SMP itself (segment CVAA, and others) has to be allocated or loaded in each subsystem in addition to the BSS.

---

## UI/Reservation Package (PARS)

The UI package which supports input and output messages for terminals logged on to reservation type applications, contains tables which are used to determine the program to be executed based on the message action code. There are two *entry* tables and one *enter* table for each application which uses the UI package (up to five applications are allowed). If the package is to be used, the user must define in segment UIIA which entry and enter tables are to be used for each application. The entry and enter tables are contained in program segment UIIB, and if required, UIIC through UIIF. The user must code the entry and enter tables specifying the action codes to be used and the programs to be activated for processing.

The entry tables convert the action codes to index values. One table converts the action codes which come in as keyboard entries, whereas the other table converts action codes which are the result of a function/action key (also referred to as the *pushbutton*) entries. These index values are subsequently used to address the enter table which contains an ENTNC or ENTDC macro for the required program.

TPF was originally designed as the operating system for the Programmed Airline Reservations System (PARS) and for the International Programmed Airline

Reservation System (IPARS) application. The UI package was that part of the application which supported the sending and receiving of messages to terminals logged on to the reservation application.

The message router requirements for the UI package are those for an application using an LNIATA rather than a resource identifier (RID) (see “Message Router Application Attributes” on page 143). If the PARS/IPARS application is required the user must specify RES=YES on the CONFIG macro and also supply the PARS user dependent variables via the USEREQ macro. The Fare Quote/Issue Ticket program product may also be included via the FQTK keyword of the CONFIG macro.

For more information about the UI package see the *TPF Data Communications Services Reference*.

---

## Unsolicited Message Support

The unsolicited message package (UMP) is a group of programs that handle unsolicited messages to terminals or logical units logged to *old* applications. A message is defined as *unsolicited* when it is not generated as a response to an input message. Unsolicited messages originate at a terminal/logical unit or in a program in a TPF processor or non-TPF processor. The destination may be either a terminal/logical unit or a group of terminals/logical units (the latter is referred to as a *broadcast* message). Routing variations are as follows:

- A specific terminal or logical unit excluding SNA applications LUs.
- All terminals attached to a specific interchange or to all logical units on a specific cluster controller.
- All terminals or logical units attached to a specific line.
- All logical units attached to a specific transmission control unit.
- All terminals or logical units hosted by this TPF system.
- All terminals or logical units logged to a specific application.
- *broadcast* is only allowed to **owned** (SNA) devices or cross-domain resources that are in session with this TPF host.

The unsolicited message package uses unsolicited message directories (see data macro CO1DR) to maintain references to unsolicited messages until they are successfully transmitted or are purged as redundant items as a result of a disposition request. The CODR directories are organized by terminal or logical unit.

When processing an unsolicited message, a reference item is created and inserted into a directory record. The directory record is retrieved using the address of the destination terminal or logical unit to calculate the needed ordinal number in the directory's record type. A reference item contains information such as:

- Message and destination terminal characteristics
- Destination address
- Origin
- Message file address, and so on.

There are 4 prime CODR records that are allocated as program records via SIP. Additional records are acquired as needed from the random pool. Initialization for prime and overflow records is handled by an unsolicited message package component program.



Broadcast messages require the creation of reference items for the addressed terminals and the propagation of the message to all other TPF processors in the network. For all broadcast messages except those routed to units of a specific application, the unsolicited message package uses the WGTA, which is created from the user-specified UAT (see "Communication Pilot Record Generation" on page 137) for routing to terminals that have a valid LNIATA or pseudo-LNIATA (3270/SDLC and NEF terminals).

Terminals logged to SNA applications, those that use SNA resource identifiers (RIDs) to identify SNA resources, are specifically excluded from receiving unsolicited messages by this package. This is because this package and the TPF system itself do not control the screen format in use by the terminals in the application. Therefore, it is not safe for this package to change the screen format from what the application itself thinks is on the screen or in the buffer of the terminal. If a user wants to send unsolicited messages to these logical units, the user must write an application program to perform this function.

In order to schedule delivery of these messages, UMP maintains the unsolicited notification list (see data macro CO3NL) containing references to each receiving terminal or logical unit. For broadcast messages, notification is sent to the terminal immediately.

For nonbroadcast messages (single messages) the destination terminal is immediately notified if the origin of the message is the same application to which the terminal is connected or the terminal is *idle*.

After receiving notification of existing unsolicited messages, the terminal operator may either temporarily disregard the notice or immediately request a display of the message.

A third functional component of UMP will, on a time initiated basis, police the unsolicited message directory (CODR) to again advise terminal users of unsolicited messages queued for the terminal. This policing function will also purge out of the system any messages not displayed after several attempts at notifying the terminal user. The purge function may also be initiated by a direct request from the terminal user.

---

## Long Message Transmitter Package

The long message transmitter package (LMT) queues output messages on file and transmits them to hardcopy terminals such as the 1977 printer/keyboard terminals, 1980-21 and 24 printers, BSC and 3284/3286 SNA printer terminals using the pseudo-LNIATA interface, and transmits them to the 1052/3215 console. LMT is also used for any of these terminals that are supported by NEF.

The following types of messages are handled via LMT:

- Unsolicited output messages requested via ZSNDU command.
- Overlength solicited output messages (that is, those which contain more characters than the terminal buffer can accommodate).
- Output message switching traffic.
- Output traffic to the 1052/3215 printer/keyboard, 1980 and BSC terminals or 3284/3286 printer terminals using the pseudo-LNIATA (PLIT) interface.

LMT requires that certain fixed file records be allocated and initialized, one for each terminal controlled by it (see “Communication Pilot Record Generation” on page 137).

---

## Message Switching Packages

Domestic and world trade message switching are discussed in this section.

### Domestic Message Switching

This package (MESW) contains the mainline programs which handle the message switching function and the overlength unsolicited application message transmission in the TPF system. The purpose of the MESW package is to provide programs which assemble, queue, transmit and log message switching messages between message switching terminals. It also receives and transmits teletype application messages. The SIP CONFIG macro parameter MSGSW is used to specify whether or not message switching is included in the system.

### World Trade Message Switching

World Trade users, those who specify USER=WTC vs. USER=USA on the CONFIG macro, are generated with the IPARS version of message switching if MSGSW=YES is specified.

---

## TPF Mapping Support

The mapping support package enables the user to format and control data streams routed between a terminal and an application program and between two application programs. As a formatter vehicle this package presents to the user program data streams devoid of device dependent code. This formatting capability is attained by user defined and package generated formats, or maps, which characterize input and output for CRTs and output for printer devices. The mapping support package additionally provides paging and scrolling multiple screen management control functions for CRT type devices.

The map file create and load program is used to create or update the map file records for the mapping support package. This program is divided into two phases:

- The initial phase comprises the offline portion, ASLOL and ASLOM, which construct the map files on tape (see data macro AS0MP). These offline programs run under the control of MVS.
- The second phase of processing consists of loading the terminal map records (see data macro AS1MP) from the tape file (see data macro AS0MP) onto the disk map files; the result is map file update or a total creation. This function runs under TPF.

The CONFIG macro parameter MAPSP is used to determine whether or not mapping support is included in the system. See *TPF Data Communications Services Reference* for complete details.

---

## Control Program User Exits

In the past, many users of TPF made modifications to the control program in order to support their unique processing needs (for example, accounting, test tools, and so on). These modifications then had to be reinvestigated and reimplemented with each migration to a new release of TPF.



In order to alleviate this problem, a set of predefined user exit points has been provided within the TPF control program. These user exit points, when activated, will cause the TPF control program to pass control to user-supplied routines, which may then perform their processing as an extension of the control program.

In order to implement this function, the user must update the source code in copy member CUSR (control program CSECT CCUEXT), assemble CCUEXT, and link-edit the control program to include the updated CCUEXT CSECT.

The user-unique code is thus isolated from the released TPF control program and need only be changed if the exit point interface has changed (for architectural design reasons) when migrating from one release of TPF to another.

The source code for copy member CUSR (as released) contains commentary only, describing the exit point interfaces.

Also, the user will have to provide an E-type program to control the activation/deactivation of the defined exit points.

See *TPF System Installation Support Reference* for more information about user exits.

---

## Global Area

The area of main storage known as the *global area* is reserved for main storage resident data records for use by both the system and application programs. The application core load and restart program (GOGO) and the special program record retrieval program (CIJH) are activated by the control program during the restart sequence to load the global areas. The records to be loaded are listed in a storage allocation table of the global storage allocator records (GOA and super GOA). The super GOA contains a pointer to the prime (first) GOA for each I-stream for each possible central processing complex (CPC) in a TPF complex. Each core allocation table (GOA) defines a set of records to be loaded by the GOGO program. MDBF users must provide GOA records for each subsystem. For a more complete discussion of the global system, see "Application Global Area" on page 40.

In addition to supplying the required GOA records via a pilot tape (see "Communication Pilot Record Generation" on page 137), the user must code the SIP GLOBAL macro (see "GLOBAL" on page 255).

To identify which of the global records/fields must be synchronized requires the use of the GLSYNC macro (see "GLSYNC" on page 258). A separate macro must be coded for each field/record, and also for each subsystem user, if MDBF is used.

See the *TPF System Installation Support Reference* for more information about globals.

---

## Data Collection and Reduction

The purpose of this package (DC/DR) is to observe by judicious sampling, the operation of the TPF system environment and report on its performance parameters and their statistical characteristics. The package is comprised of an online subpackage referred to as data collection (DC) and an offline (MVS) subpackage called data reduction (DR).

DC is the online data collection subpackage. It collects and records real-time TPF system performance information. It observes system performance by collecting predetermined indicators, at fixed time intervals, through interaction with the control program at the bidding of its own driver which is operator-initiated.

DR is the offline data reduction subpackage which runs under MVS. It reads, edits, and arranges the recorded performance information. The data is mathematically analyzed to formulate it in operationally meaningful terms and the results are described in standard statistical measures via printed reports.

The SIP macro DATACO is used to specify values used by both the data collection and reduction packages. This macro is optional (all default values may be accepted) and must only be coded once in the SIP Stage I input for the BSS. These values will become part of the data reduction pre-compiler segment (JPC0) that is used in compiling the other data reduction segments.

Additional information about DC/DR may be found in the *TPF System Performance and Measurement Reference*.

---

## Database Reorganization (DBR)

Once the TPF system has been initially generated, cases occasionally arise in which the configuration of the fixed file areas on disk need to be expanded or altered. This requires juggling the present fixed file records in order for data record accessing be compatible with the new layout.

The database reorganization package provides facilities to reload all records affected by a change to the DASD file configuration.

MDBF users of the HPO feature may use DBR to reorganize the fixed file records of a particular subsystem as opposed to reorganizing all of the fixed file records in the system.

For more information see Database Reorganization in the *TPF Database Reference*.

---

## Recoup

Pool addresses can become *lost* to the system in a number of ways (for instance, application program errors or system failure). The file pool recoup program is provided to recover these lost but still usable pool records. The lost records are made available to the system for further usage. The recoup program must interface closely with the application environment to determine which records are valid. As a result, the user must allocate and initialize the recoup descriptor records which define where the package can find the pool addresses within the user's application fixed file records.

A by-product of the file pool recoup program is information which identifies application programs which may be responsible for not returning file pool records to the system.

MDBF users of the HPO feature may use the recoup package within each subsystem and as a result, the descriptor records must be uniquely defined and allocated for each subsystem.

For more information, see *TPF Database Reference*.

---

## File Capture and Restore

The capture and restore packages enable the user to capture all or part of his system at a given condition and restore the system to a predetermined condition in the event of a hardware or software failure. MDBF users of the HPO feature may capture/restore individual subsystems. For additional information, including a detailed explanation of the SIP macros discussed below, see the *TPF Database Reference*.

The SIP requirements for capture/restore include these SIP macros:

1. The DDCCAP macro is used to establish the disk device control table for capture/restore. Parameters may be submitted for up to four device types. If coded, parameters for at least one device must be entered.
2. The LOGCAP macro is used to specify information relating to logging/exception recording (LOG/XCP) processing. Values are entered to specify the online module ranges to be assigned to each of the LOG/XCP exception recording tapes.
3. The RESCAP macro is used to specify error counts, channel utilization percents, and other variable data pertaining to capture/restore. It is also used to input the channel, control unit, and relative speed of available tape units.

All these macros are required by SIP if the capture/restore facility is desired.

---

## Loaders

SIP stage II will create and list the required MVS Job Control language (JCL) and input to run the offline TPF loader for the loader general file (TPFLDR with PARM='ALDR'). Users must actually run this MVS job. The output of SIP is placed into a named PDS member LOADDECK of the source library for access by the user.

**Note:** SIP also creates and lists the load list of E-type programs. This is placed in PDS member PARSxx, where xx is the version number of the current release for domestic customers or where it is the version number of the World Trade load list. (USA or WTC is specified on the USER parameter of the CONFIG macro).

Once the online system has been loaded as a result of the load to the loader general file, the user will often find it necessary to either reload certain programs or to load additional programs. Two facilities are provided: the E-type loader and the auxiliary loader. Associated with these program loaders, there is also a system data loader that is used to load data records (pilots) to the online modules.

## TPF General File Loader

The TPF general file loader is designed to load the online modules with all the information required before the control program restart sequence (initial IPL) can be executed. The general file loader consists of two segments: the general file loader offline segment and the general file loader online segment.

The general file loader offline segment (ALDR), operating under the control of MVS, creates an IPLable loader general file. The control program is loaded to this general file. Type 'E' application and system programs are link-edited by the loader using the system allocator record and the TPF linkage editor prior to being loaded to the general file. The JCL and data required to create the initial loader general file is generated by SIP.

Once the offline segment has successfully loaded the loader general file the latter is IPLed. Now the system is under the control of TPF.

The IPL program loads programs to main storage from the loader general file as a result of the IPL. The control program, keypoints, startup programs, FACE table, core-resident application programs, etc., and the general file loader online segment (ACPL) are brought into main storage. If this is an online module load operation, the general file loader online segment takes control at this point and loads programs to the online disk modules using the loader general file as input.

The general file loader can also *patch* programs as it is loading them. By the appropriate choice of control cards the general file loader will load any or all items required.

## E-Type Loader

The E-type loader allows a user to load an unlimited number of E-type (or real-time) program segments onto the online system while the system is operational in any state. Loadsets, which are groups of programs to be loaded, are created offline and then passed to the online system using a tape, general data set, virtual reader, or user-defined device. The offline process is run by submitting JCL that runs the offline E-type loader segment (OLDR). The E-type loader package requires that #OLDx (where x corresponds to a program area) fixed file records be allocated for use as online directory records and program records. For example, if #PROG4 programs are being used, #OLD4 records must be allocated.

## Auxiliary Loader

The auxiliary loader allows a user to load the control program, keypoints, E-type programs, and the E-type start-up program segments to an inactive image of TPF. The TPF system can be in any system state. The offline auxiliary loader creates a TLD tape, which contains a tape directory and the programs to be loaded. The auxiliary loader requires the allocation of certain fixed file records for use as an online directory.

## Data Loader

The data loader loads pilot system data, which are located on the SDF tape, to the online disk modules. Pilot tapes are created by either the system test compiler (STC) or the SNA table generation process (see “System Test Compiler” on page 158 and “Communication Pilot Record Generation” on page 137).

Use the ZSLDR command to activate the data loader and load the pilot tape. If the ID of the pilot tape is N, the TPF system can be in any state. If the ID of the pilot tape is not N, the TPF system must be in 1052 state. You can also activate the data loader after the general file loader online segment has been loaded (see “TPF General File Loader” on page 155).

**Note:** If the pilot being loaded contains global records, you must re-IPL the system to get these records into main memory.

See *TPF System Installation Support Reference* for more information about loaders. See *TPF Operations* for more information about the ZSLDR command.

---

## Pool Directory Generation and Maintenance

Pool directory generation and maintenance functions generate and update directory records that are used by online file pool management programs (GFS and RFS). User input to SIP (see “Pool File Storage” on page 68) creates the information necessary for the GFS and RFS programs. See *TPF Database Reference* for more information about pool directory generation, pool maintenance functions, and online file pool management.

---

## Program Test Vehicle

The program test vehicle (PTV) is capable of executing two different test phases under the control of TPF. These are *package* and *system test* and they are referred to by phase III and system test vehicle (STV) modes. With PTV, a user's application programs can be executed at various levels of testing while executing in a real-time TPF environment. Additional information about PTV testing may be found in *TPF Program Development Support Reference*.

The generation of a system with PTV support requires the user to code the PTV keyword on the CORREQ macro. Three choices are available:

- NO**     There will be no PTV support (the default).
- YES**     PTV support for both levels of testing.
- STV**     PTV support, but only for system test mode.

**Note:** The specification of either YES or STV will also cause an indication in keypoint record 'A' (CTKA) that the system is in *test mode*. This same indication is also set on via the specification of VM=YES on the CONFIG macro.

In addition to the SIP generation requirement for PTV, the user must also perform other tasks for the proper generation of PTV support. These tables are related to the type of communications support that the user is employing (see “Data Communications Support” on page 123).

For 3705 EP communications support, the user must define the 3705 EP lines which are to be used by the application programmers when coding test unit input for PTV phase III and STV modes via the system test compiler (STC).

1. PTV high-speed input may be from ALC, BSC, 3270 local, and/or SLC pseudo lines. The changes and generation considerations necessary for PTV 3705 EP support are also required for ALC lines supported by ACF and NEF, but NEF input also requires PTV/SNA generation requirements (see PTV considerations below).
2. The user chooses whether to use *real* lines or to define for PTV a completely simulated 3705 EP communications network for its use. The difference is that if real lines are used, there can be no real input from those lines when executing PTV. If the choice is to use a *simulated* network, then the PTV user must generate that network all the way down to the terminal level (see “Communications Record Generation” on page 125) as if it really existed. Note, extreme care must be exercised by the system operators not to try to actually activate the simulated lines.
3. Prior to executing SIP, the user must physically update a PTV segment to define these lines:

- a. ACP.SRCE.OL.RELv PDS member NPTV, which is *copied* by CP segment CCUTIL and assembled by SIP, contains the simulated line number table that is located at label 'BRSHST'. Each is a two-byte entry where the first byte contains bit indications about the line and the second byte contains the actual symbolic line number (SLN) to be used. The listing commentary of NPTV explains how to code the bit indications. Entries not in use must be coded as a X'00FF'.
  - b. The released version of NPTV contains entries for the "TPF Development" database and must either be physically updated prior to SIP Stage II or the entries patched into CCUTIL prior to execution. For PSEUDO SLC lines, the line number and the associated link must be defined in NPTV so that PTV can make use of that line.
  - c. Line X'01', which represents prime and RO CRAS, should not be put into these tables. Input from the prime CRAS (LNIATA of X'010000') via STC may be coded without this table entry. Note that input of commands via STC from other than the prime CRAS can also be accomplished by coding input from a simulated alternate CRAS terminal which has been defined in the database and/or entered via the ZACRS command prior to PTV execution.
  - d. At least one entry must be in each of the two tables.
4. In addition to the updates required for PTV itself, users of PTV/STV mode and 3705 EP communications must also enter all of the simulated terminal addresses being used into tables located within the PDS member BMP0 of the post processor (see "Diagnostic Output Formatter" on page 159). This is required to perform *terminal simulation* by the offline formatter.

---

## System Test Compiler

The system test compiler (STC) is a multi-purpose tape-file generator of both the PTV test unit tapes (TUT) and for standard data file (pilot) tapes (SDF). It is also available to create pilot tapes containing user required application data records, including global records. STC is executed under the control of MVS. Additional information about how to code STC input, plus operating instructions for the package, may be found in the *TPF Program Development Support Reference*.

As part of a user's generation procedure, the user will have to load and perhaps update two files associated with STC. These are the SDMF and the DRIL files. The release tapes for TPF contain both of these files but they should be reviewed and may need updating prior to the execution of STC.

1. The data record information file (DRIL) contains an entry for all system data records and message formats. Use of this file allows input to STC to be symbolic vs. absolute.  
To use symbolic addressing for user created data records requires that the DRIL file be updated with a description of each new data record. The updating and loading of DRIL is described in the *TPF Program Development Support Reference*. The DRIL file itself is described in data macro DR1IL.
2. The standard data/message file (SDMF) goes a step further than DRIL and allows not only symbolic addressing but the actual placement of data within the records defined. For example, fixed file records could be placed into SDMF with the record ID preset. As a result, STC users would not have to initialize the record ID. Any field within a record can be pre-specified in SDMF. The STC user has the opportunity to override the SDMF information. This capability is useful when generating large volumes of records. The updating and loading of SDMF are described in the *TPF Program Development Support Reference*. See data macro SDFPF for a description of the file.



---

## Diagnostic Output Formatter

The diagnostic output formatter is really two packages. One of these operates offline under MVS and is known as the *post processor*. The other operates online under TPF and is known as the *in-core dump formatter* (ICDF). SIP automatically includes both of these packages in the generated system. For more information see the *TPF Program Development Support Reference*.

The function of these packages is to format TPF diagnostic data on a printer in a form that is easily readable by the programmer for the purpose of debugging. Normally, all TPF *log* type output is directed to the RTL tape ("Tape Support" on page 52). This tape is then read and formatted by the offline post processor.

The types of output logged and formatted are:

- System error dumps
- Program test vehicle (PTV) output
- Real-time trace (RTT) output
- Selective file dump and trace output (SFDT)
- CCP trace output.

Information about all of these diagnostic test tools may be found in the *TPF Program Development Support Reference*. Other data logged to the RTL tape is processed by other offline packages (for example, SNA's PIU trace).

ICDF, the online portion of the diagnostic output formatter, is designed to only format system error dumps of TPF's main memory directly to a printer. The address of the printer to be used is specified at SIP time on the APRNT keyword of the CONFIG macro. This facility can only be used in a *test* system. The ZASER command is used to direct output to the online printer via ICDF or to the RTL tape which will be processed by the offline post processor. All other diagnostic output, normally directed to RTL tape, will still be directed to it when ICDF is used to format system error dumps.

**Note:** The generated system is set up to use the RTL tape.

One of the functions of the offline version of the diagnostic output formatter is to perform terminal simulation on the input/output message stream that has been logged by PTV to the RTL tape. These messages are printed by the post processor in the same form that they would be seen on the actual communications terminals. In addition, errors in the message text, as a result of terminal characteristics or requirements, are detected and reported.

Terminal simulation is provided for the following communication devices:

- 2915 and 4505 CRTs
- 1977 printer/keyboards
- 1980-21 printers
- 1980-24 ticket printers
- 83B teletype terminals
- 3270 local.

For PTV phases I and III, the terminal simulator programs of the offline diagnostic output formatter know which messages are for which terminal types based upon

user input to STC that identifies the terminals by address (LNIATA). This information is passed by PTV to the post processor via the RTL tape.

For PTV/STV mode, a preset table in segment BMP0 of the offline formatter is used to identify specific 3705 EP, 3270 local, and NEF supported terminals by terminal address.

The user must physically update each of the tables within segment BMP0 to reflect the terminals and terminal types that will be used for input during PTV/STV mode testing. The tables must also contain the addresses of any printers that are to receive output and are on *simulated* lines during the STV test. The use of a simulated communications network during PTV testing is discussed in "Program Test Vehicle" on page 157. The entries in the BMP0 tables should also be consistent with the terminal type entries made in the UAT.

There are six tables within segment BMP0, one for each of the supported communication device types, except for SNA devices. The 3270 terminal table is further subdivided into eight subsections based upon 3270 model type.

1. Each table entry, except for 3270 devices, consists of four bytes:
  - a. The first byte is a X'00'.
  - b. The next three bytes are the terminal address (LNIATA). Note, for NEF supported terminals, the pseudo terminal address should be used.
  - c. The entry for the teletype (83B) terminals, which are addressed by only a one-byte low-speed line number, should be placed into the table as X'0000LN'.
2. For 3270 terminals, each entry consists of six bytes:
  - a. The first byte, instead of being a X'00' as it is for non-3270 terminals, is a bit combination that is used to identify the 3270 model type. An explanation of the bit settings appears in the BMP0 listing commentary.
  - b. The next three bytes are the LNIATA of the 3270 terminal. Note, 3270 local terminals should appear as X'LN0000'.
  - c. 3270/SDLC pseudo terminal addresses should not appear in the table.
  - d. The next two bytes for each entry is the EBCDIC control unit and device address that is used by the 3270 **copy** command.

**Note:** The CU/DV address should be coded as two blanks, X'4040', for 3270 local terminals since they do not support the **copy** command.

The list below identifies the labels used within BMP0 and the terminal type that they represent:

- BMPAT2 - CRTs (2915s and 4505s)
- BMPAT4 - 1977s
- BMPAT6 - 1980-21s
- BMPATA - 1980-24s
- BMPAT8 - teletype 83Bs
- Local 3270s: (see Note 1)
  - BMPATC - 3277 model 1s
  - BMPATE - 3277 model 2s
  - BMPATK - 328x model 1s (see Note 2)
  - BMPATM - 328x model 2s (see Note 2)

**Note 1:** 3270 support within the terminal simulators is obtained via the use



of conditional assembly code; the 'AIF' statement. As a result, the 3270 tables are not present unless the user has requested one of the various 3270 supported protocols (e.g., 3270 local or BSC), except for 3270/SDLC.

**Note 2:** These tables are for all of the supported 3270 printers except for the 3284 model 3. Special mention should be made of the relationship of this package to the ISAM data set created by the SNA table generation process. When formatting either PTV or RTT output, it must convert the logged SNA resource identifier (RID) to a nodename. The ISAM data set, which is used to perform this conversion, must represent the same online TPF/SNA data base that has been loaded to the TPF system. In addition, the terminal simulator uses this data set to obtain terminal/device characteristic information. If, for example, a resource has been added to the online database, but an old ISAM data set was used, the converted RID would not identify the correct SNA resource.

---

## Real-Time Disk Formatter

All of TPF's online and general files must be properly formatted before they are loaded with programs and data records. The disk formatter (FMTR) runs under the control of MVS and is used to format a disk pack for use in the TPF system.

SIP Stage II will create the JCL to format the required loader general file.

With ISO-C support, SIP splits the L1A2E job into L1A2E and L1A3E during Stage II. In this case, you need to modify your JCL. Following is a sample of the JCL produced by SIP.

```
//L1A2E EXEC PGM=FMTR40,REGION=8192K
//STEPLIB DD DSN=ACP.LINK.REL40.BSS,DISP=SHR
//DUMMY DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//FMTDD2 DD SYSOUT=A
//FMTPSN DD SYSOUT=A
//FMTDD3 DD DSN=GNFLBSS,UNIT=3380,
//          VOL=(PRIVATE,,,SER=SA0999),
//          DISP=(NEW,KEEP),SPACE=(ABSTR,(1,1))
//FMTDD1 DD *
FMT      L      N 0000001 0000001 338L
FMT      END
/*
//L1A3E EXEC PGM=FMTR40,REGION=8192K
//STEPLIB DD DSN=ACP.LINK.REL40.BSS,DISP=SHR
//DUMMY DD DUMMY
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//FMTDD2 DD SYSOUT=A
//FMTPSN DD SYSOUT=A
//FMTDD3 DD DSN=GNF2BSS,UNIT=3380,
//          VOL=(PRIVATE,,,SER=SA0999),
//          DISP=(NEW,KEEP),SPACE=(ABSTR,(522,2))
//FMTDD1 DD *
FMT      4      N 0000002 0003413 3380
FMT      END
/*
```

Stage II will only create and list the required MVS Job Control Language (JCL) and input to initialize and format the first module of each device type for the online files. It is up to the user to actually run these MVS jobs. It will also be necessary to

modify the VOLSER= parameter of the JCL for modules other than the first module of each device type. The output of SIP is placed into PDS member FMTRDECK of the source library for access by the user.

---

## FACE Driver and Offline Interface

TPF provides an offline (MVS) program, DFAD, as an aid to the user in managing the unique TPF database. It has two functions:

1. Provide a facility that will list all record type and ordinal number combinations of fixed file records along with their associated file addresses.
2. Provide a *bridge* between user written offline programs and TPF's FACE program and FACE table in order that these programs can dynamically determine the file address of a record given its record type and ordinal number.

Information about how to use this program may be found in the *TPF Database Reference*.

---

## System Clocks

TPF operates using the time-of-day clock (TOD) to manage various system functions such as subsystem clock update. Upon initial IPL of TPF, the system operator will be requested to set the correct TOD value if the TOD clock is not operating.

Users of the HPO feature and the loosely coupled facility are required to install either the clock synchronization RPQ or a Sysplex Timer (STR), so that all processors in the LC complex will run their TOD clocks with the same value. The synchronization is done upon system restart.

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

The TOD clock itself is not used for application programs. TPF provides software clocks for their use. These software clocks are managed as a *delta* to the TOD clock. The initial values of the various software clocks are provided via the SIP CLOCKS macro.

The user must provide the calendar starting time of the system, along with the local and Greenwich Mean Time (GMT). For HPO feature users of the multiple database function (MDBF), the user must provide a CLOCKS macro for each subsystem being generated, plus an additional parameter (DELTIM) that specifies the subsystem difference (delta) from the system clock.

**Note:** The BSS clocks also operate on the basis of a delta from the TOD clock even without the use of MDBF.

---

## TPF Collection Support

TPF collection support (TPFCS) is a database manager service that enables application programs running on TPF to create, modify, and access collections. To provide support for this function, two fixed file records from the IBM miscellaneous 4-K record type are allocated for use to contain the prime and shadow TPFCS control records, respectively. These records are used to maintain information about the user data stores defined to TPFCS. See *TPF Concepts and Structures* for more information about record types, ordinal numbers, and data stores.

TPFCS uses the following:

- 4-K long-term pool records for all long-term persistent data records it creates.
- 4-K short-term pool records for all short-term persistent data records it creates.
- Private heap area storage of the ECB with overflow to short-term pools for temporary collections.

(See “Creating the RIAT” on page 184 for information about the SPRIAT macro and for the definition of the default record ID values used by TPFCS.)



---

## Part 3. Generating the TPF System



---

## Introduction to SIP

The system initialization package (SIP) is that part of the generation process that creates a TPF system tailored to both the hardware configuration and data requirements of an installation. SIP uses the macro assembler language, so a working knowledge of the assembly process and its job control requirements is essential.

SIP provides a framework of macros to define the hardware configuration, data file requirements, and communication facilities for your installation. During SIP, the system is customized according to these specifications. The system that results from this generation will be composed of those program items required for your configuration, other programs from IBM distribution libraries, and your own installation's programs.

The new system is generated in two SIP stages as shown in Figure 26 on page 169. Before stage I, the FACE table generator is run to begin creating a database. During stage I, macros are coded that describe your configuration and data requirements. They are assembled and analyzed for errors, and they produce a job stream. It is essential for the correct job stream to be generated.

During stage II, this job stream is processed to generate the module libraries that comprise the new system. This new system contains some modules supplied by IBM and, optionally, modules that you supply.

---

### Stage I: Producing the Job Stream

After a period of planning, the desired configuration SIP stage I macros are coded to define the database, the communications network and I/O devices, system operation, main storage requirements, and so on. The FACE table generator program (FCTBG) is used to read database definitions in the stage I definitions and produce a FACE table that mirrors the definitions in the stage I material. Next, the stage I deck is assembled. If any errors are found during assembly, error messages (in the form of assembler MNOTEs) are issued and the job stream is not produced. The errors must be resolved satisfactorily before the job stream is produced. The system generation error messages are found in *Messages (System Error and Offline)*.

If no errors are detected, a job stream consisting of job control language (JCL) statements and source code is produced. During stage II, this job stream is run under MVS control to select and process modules from the IBM distribution libraries and optional user libraries to form the new system.

---

### Stage II: Processing the Job Stream

In stage II, the job stream produced during stage I is processed as follows:

- Configuration-dependent macros, keypoints, offline programs, and real-time programs are placed into SIP source libraries.
- Selected modules are assembled.
- Linkage editor processes selected offline modules.
- Loader general file module is initialized and formatted.
- The operational program list (PARS list) for the release is generated and assembled.

- System load deck is generated and listed according to input from stage I.
- JCL to initialize and format the online modules is created.
- Selected programs are run to complete the initialization process.

A complete listing of stage II functions by job and step is shown in Appendix C, "SIP Stage II Job Summary" on page 473.

The SIP stage II job stream must be run in the exact sequence that it was produced from SIP stage I. This is because of the sequential dependency of the individual stage II jobs and steps. Appendix C, "SIP Stage II Job Summary" shows the interdependencies for the various jobs and steps. In general, stage II error recovery requires complete reexecution of both SIP stage I and stage II. Individual jobs and steps may be reexecuted if the problem that caused the original failure is not a result of stage I input errors (for example, stage II hardware errors).

The completion of stage II results in a system that reflects what was specified in the SIP macro instructions. For example, functionally dependent segments that are not required are excluded from the system.



## Stage I

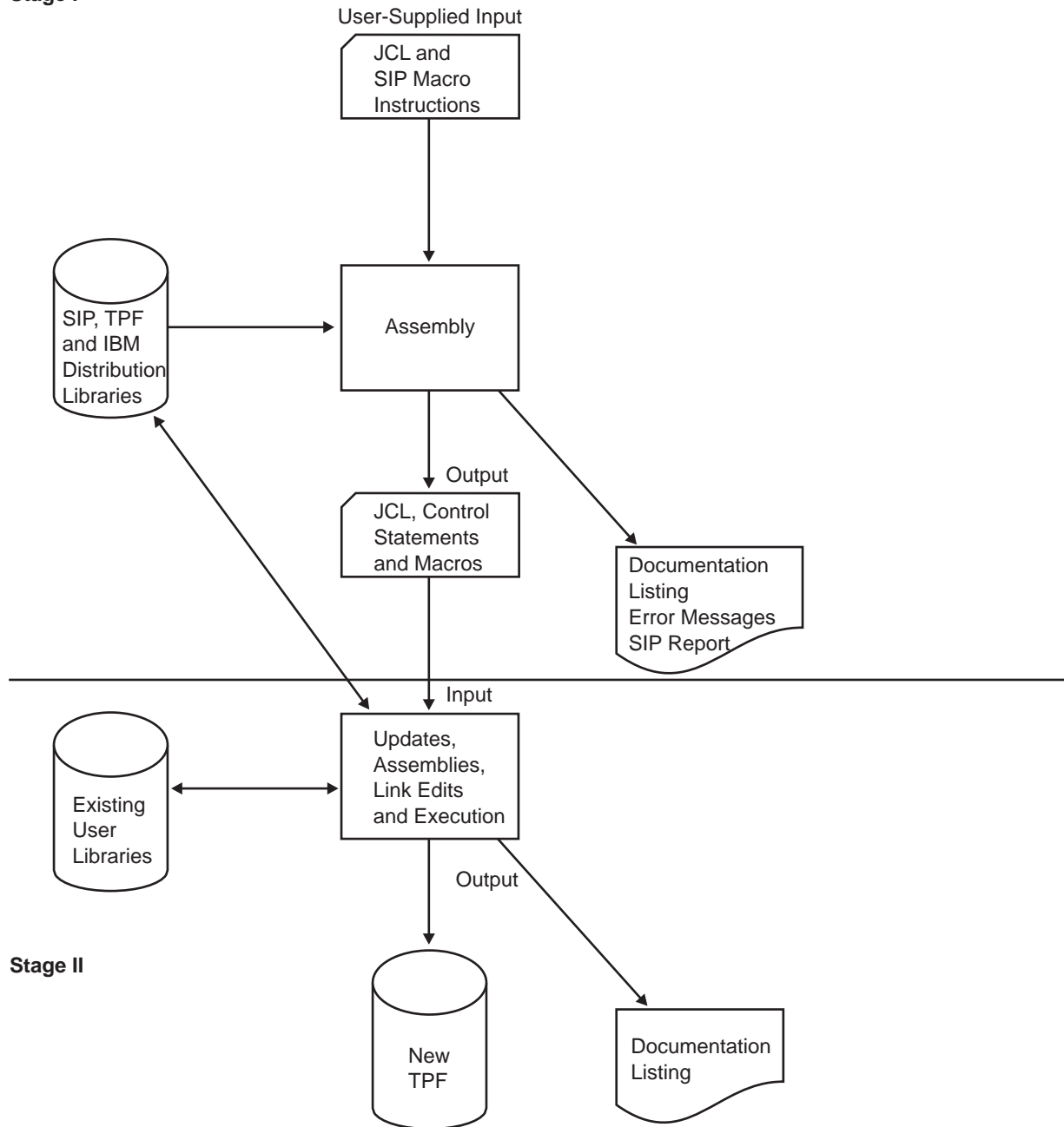


Figure 26. Functions Provided by SIP

## SIP Functions

SIP performs the following functions for generating a system:

Before stage I, the FACE table generator reads the RAMFIL statements in the stage I deck and produces the FACE table as well as several other tables.

## Stage I

- SIP provides a set of comprehensive user macros for specifying user dependent variables. These macros appear in, Stage I: Coding the SIP Macros starting on page 207.
- SIP provides a set of diagnostics to analyze, and in some instances, correct parameters coded on the user macros. All diagnostic messages are in the form of MNOTEs. These diagnostic messages are categorized by their user macro code and SIP diagnostic number. See *Messages (System Error and Offline)*.
- SIP produces a system generation report at the end of the stage I process. This report provides the user with a detailed breakdown, by macro, of the input parameters supplied. In addition, calculations performed by SIP will also be reported. See “SIP Report” on page 329 and also “Sample SIP Stage I Output - SIP Report” on page 472.
- SIP stage I produces a system generation job stream in 80-column card image format called the stage II input. It is the stage II job stream that must be run to actually create the system.
- SIP creates procedures to assemble offline programs, control programs and real-time modules. Additional JCL is provided to compile online and offline C segments.
- Some SIP stage I macros allow you to specify your own program and macro libraries for inclusion in the created stage II input stream in addition to the released program and macro libraries. For example, RT and CRT correspond to URT and CUR, respectively.

**Note:** ISO-C support does not permit users to specify their own program and macro libraries.

- SIP creates system-dependent macros at stage I time using the inputs supplied to the user macros.
- SIP creates the source code for both the general file and online keypoints, in addition to the source code for the tape label programs and certain message router program segments.
- The operational program list is created during the stage I process.
- SIP stage I creates the JCL and job stream necessary to format the loader general file, the online modules, and also creates the load deck to be used for the initial system load.
- SIP stage I will create TPF subsystems in a multiple database function (MDBF) environment. Each subsystem is a unique stage I run. SIP will use a LOOK BACK facility in the form of a macro called BSSSET. This macro will describe to the nonbasic subsystem generation the functional features generated into the basic subsystem. This is one of the reasons why the basic subsystem generation must be complete before starting the nonbasic subsystem generations. SIP creates the necessary support for the loosely coupled (LC) environment.

## Stage II

- Stage II places the dependent macros created during stage I into macro libraries.
- Stage II places the procedures to be used to assemble offline, control program, and real-time modules into a procedure library. The JCL to compile C language segments is also inserted into a procedure library.
- Stage II places those programs created during the stage I process into the program libraries.

- Stage II assembles, compiles, and link edits all programs determined by stage I to be necessary for the complete custom system. These include:
  1. All programs categorized as offline, such as STC and Post Processor.
  2. All programs categorized as keypoints, such as CTKA and CTKB.
  3. The control program.
  4. All programs categorized as real-time, including user programs.

**Note:** Programs shipped as object code will not be assembled but will be link-edited and executed if necessary. See “Creating Program Tables” on page 178 and “Modifying the Program Tables” on page 181.

- Stage II places in data sets the inputs to other program segments to be executed at a later time than the initial stage II generation, such as the application name table definitions to be used by the offline table generation program (OSTG), and the global definitions to properly execute the GTSZ program.
- Stage II initializes and formats the loader general file to be used at system start. In addition, it produces JCL to initialize and format the online disk modules.

---

## Before SIP

### Unpacking

- Before executing SIP, unpack the release tapes and perform certain library functions described in the Program Directory document distributed with the tapes. These library functions and required data sets are summarized in the following paragraphs.

**Notes:**

1. vv refers to release version number.
2. yyyy refers to the subsystem name.

**ACP.SIPGEN.RELvv**

This library contains SIP stage I macros that must be coded. In addition, the SIP skeletons that are used for generating the configuration-dependent source code are contained in this library. This particular data set is referenced by SYSLIB data definition statements during execution of SIP stage I (see “Sample SIP Stage I Input” on page 351 in Appendix B, “Sample SIP Input and Report”).

**ACP.SYMACRO.RELvv and ACP.SYSRCE.RELvv**

These data sets must be cataloged data sets and space must be allocated by the user for them. Upon completion of SIP stage II, they will contain source code reflected by stage I inputs to the various SIP skeletons. These data sets contain the SIP-generated macros, keypoints, tables, and records created by executing the stage II JCL stream.

The SYMACRO data set also contains the macro SYFCTB, which is created by the FACE table generator. The SYMACRO data set should be referenced by the SYSLIB data definition statement during the execution of SIP stage I so that SYFCTB is picked up. The SYMACRO data set for the BSS should also be referenced by any non-BSS subsystem stage I assemblies so that the BSSSET macro can be picked up.

**ACP.OBJ.RELvv and ACP.LINK.RELvv**

These data sets must be cataloged data sets and must be allocated by the user. These data sets will contain the assembly or compiler output

object modules and link-edit modules that will be created by executing the stage II JCL stream. For more information, see Figure 27 on page 177.

#### **BDF.V1R1M3.BDFOBJ1.yyyy and BDF.V1R1M3.BDFLNK1**

These optional TPFDF data sets must be cataloged data sets and must be allocated by the user of the TPFDF product. These data sets will contain the assembly or compiler output object modules and link-edit modules that will be created by executing the stage II JCL stream. For more information, see Figure 28 on page 177.

#### **ACP.CLIB.RELv**

This library is dedicated to shared library call linkage stubs. Note that there are no associated source files for the object files in ACP.CLIB.RELv. The object files contained in this library are dynamically produced by the library interface tool.

#### **ACP.STUB.RELv**

This library contains DLM linkage stubs, which are generated by the DLM call stub generator.

#### **ACP.IMPORTS.RELv**

This is a definition side-deck SYSDEFSD data set that must be a cataloged data set and must be allocated by the user. This data set contains functions and variables exported by a dynamic link library (DLL). For more information, see Figure 27 on page 177.

#### **ACP.SALTBL.RELv**

This must be a user-allocated cataloged data set that will contain the output of the system allocator, which is run during SIP stage II. World Trade users of the IPARS application must unpack the IASC-supplied IPARS source libraries. Space must be allocated for this data set. For more information, see Figure 27 on page 177.

**Note:** SIP output library requirements, as defined in Appendix F, “SIP Stage II Data Sets”, are satisfied through the use of the allocation information provided in the Program Directory. In a multiple database function (MDBF) environment, the SIP output libraries contain an additional name qualifier, the subsystem name (for example, ACP.LINK.RELnn.ssname).

Allocate the library data sets with the JCL shown either in Figure 27 on page 177 or in the *Program Directory*.

- Compile and link the FACE table generator using JCL “Compiling the FACE Table Generator” on page 188.
- Code a SIP stage I deck and SALO input.
- Create a RIAT source deck.
- Run the FACE Table Generator using the JCL shown in “Running the FACE Table Generator” on page 195.

## **System Requirements**

This section outlines the hardware and software requirements that are needed for running SIP.

### **Stage I**

SIP stage I may be executed on any system with MVS assembler capability, subject to the following limitations:

The HLASM assembler is required for SIP stage I.

The amount of time needed to complete a stage I process is variable. It depends on the machine processor model, the amount of work being done on that machine, core size, version of the assembler, channel utilization, and configuration. It is recommended that the stage I process be done during a period of time when the system work load is at a minimum. A system that can be totally dedicated to the stage I process will speed up the process considerably.

Function		Requirements (Minimum)
Input	SYSLIB	One DASD containing the concatenated data sets ACP.SIPGEN.RELv, and ACP.MACRO.RELv, and ACP.SYMACRO.RELv.yyy. <b>Note:</b> In a multiple database function (MDBF) environment, data set ACP.SYMACRO.RELnn.BSS must be concatenated to provide the LOOK BACK facility required for nonbasic subsystem generations.  ACP.SYMACRO.RELnn.yy (where yy is the name of the subsystem currently being generated) must be concatenated so that member SYFCTB can be picked up from the FCTBG.
Input	SYSIN	One 80-column sequential data set containing the coded SIP stage I macros.
Output	SYSPUNCH	Either 1 DASD or 1 tape drive for stage I output.
Output	SYSPRINT	One printer for assembly listing.
Work space		DASD devices containing space for assemble work areas.

**Note:** Because of the great number of modules being assembled, the region, time, and workspace parameters should be coded at maximum values for performance considerations.

## Stage II

SIP stage II may be executed on any system with MVS capability subject to SIP user macro GENSIP operands and the following requirements:

Function	Requirements
<b>Operating System</b>	MVS/SP Version 1
<b>PL/I Compiler</b>	O (IEL0AA) — XA Version
<b>C/C++ Compiler</b>	One of the IBM C or C++ compilers supported by the TPF system. See <i>TPF Migration Guide: Program Update Tapes</i> for more information about C/C++ compilers.  C and C++ programs that you want to trace using C function trace must be compiled with the TEST option of one of the family of C/C++ compilers that are supported by the TPF system.  The C/370 compiler is supported for migration purposes only.  SIP will create the JCL procedure SIPEDCC to call MASM. MASM can execute multiple C and C++ compilations.

**Note:** Before using SIPEDCC, check it to ensure it uses the data set naming conventions of your installation. Update it as necessary.

Alternatively, copy the EDCC procedure used at your installation and name it SIPEDCC. Change PGM=EDCCOMP on the EXEC statement to PGM=MASMvv (where vv is the version code).

<b>Assembler</b>	HLASM
<b>Time</b>	Preset by stage I for each stage II job and step.
<b>Input stage II job stream</b>	The sequential data set as defined in stage I output as SYSPUNCH.
<b>Release Source Libraries</b>	One or more DASDs containing the release source code data sets.
<b>User Libraries</b>	One or more DASDs for user libraries (USRCE, USMAC, USOBJ parameters) that must be cataloged data sets. User libraries must be blocked to 400 or less. (BLKSIZE=400)
<b>Loader General File</b>	One DASD (LGFDV). See <i>TPF Migration Guide: Program Update Tapes</i> for supported DASD types.
<b>Output SYSPRINT</b>	One printer for stage II output.
<b>LISTAPE parameter</b>	One tape drive for MASM assembly output, if requested (specified as "UNIT=TAPE").
<b>ISO-C library for function stubs</b>	One stub for each C library function.
<b>DLM stub library</b>	One stub for each program called by a DLM.
<b>Object Library</b>	One DASD for release object library data set.
<b>Notes:</b> <ol style="list-style-type: none"> <li>Any segments or modules that are shipped only as object code must already be present in the object data set before the execution of stage II.</li> <li>IBM encryption modules BQKCIPH or BQKDES, if installed, before executing SIP stage II (see "CONFIG" on page 219).</li> </ol>	
<b>Load Module Library</b>	One DASD for load module library data set.
<b>Note:</b> As a minimum, only 2 devices are needed if the released source, and object and load module data sets are allocated to the same DASD. It would even be possible to allocate the user libraries to this same DASD, if space is available.	
<b>System Job Queue</b>	SYS1.SYSJOBQE data set of an MVS system must be defined large enough to hold the largest SIP stage II job.
<b>System Data Sets</b>	SYS1.MACLIB and SYS1.AMODGEN must be available for use during stage II processing. These

data sets contain macros that are needed for assembling certain TPF programs.

**Output Queue (SYSPRINT)** Output from stage II is directed to the printer as follows:

<b>Offline programs</b>	A queue
<b>Control program</b>	K queue
<b>Real-time programs</b>	D queue
<b>Keypoints</b>	A queue
<b>TPF job execution</b>	A queue

Unless the OUTCL parameter is used in the GENSIP macro.

**Note:** If the LISTAPE parameter of the GENSIP macro is coded, error free assemblies will be directed to the tape, but listings with errors will be directed to the appropriate queue.

---

## Preparing for Stage I

Before SIP stage I can be executed, perform the tasks as outlined in the following paragraphs, depending on your system needs. Special attention should be given to “Creating Program Tables” on page 178 and “Creating a Site System Allocator” on page 183.

## Stage II Job Card

During stage I execution, a job card will be constructed for each stage II job. This is done by accessing a member (SPJOB) of the ACP.SIPGEN.RELVv data set. This member must be modified according to your needs by adding such items as accounting information or other special functions.

**Note:** Use extreme care whenever updating this member because variable symbols are assigned during stage I.

## SYSPUNCH Output

If the output of stage I is going to tape, an output tape must be initialized, labeled and defined in the stage I JCL SYSPUNCH data definition (see example in “Sample SIP Stage I Input” on page 351).

### LISTAPE Output

If the LISTAPE parameter of the GENSIP macro is coded (for tape output of multiple assembler), output tapes must be initialized and labeled to conform to the values of the LISTAPE parameter of the GENSIP macro. You may specify volume serial numbers or scratch for each tape. These tapes should be standard label tapes.

### ICKDSF Data Set Initialization

ICKDSF allocates and initializes a whole disk storage volume. Once the disk pack is scratched, data sets are allocated on it for the various TPF required files. Special attention should be given to the allocated block size of the FACE table to be generated. The following JCL describes the necessary steps.

1. The following JCL describes the ICKDSF run for a volume that is not online to MVS.

```
//INITDISK EXEC PGM=ICKDSF
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
INIT UNITADDRESS(xxxx) VTOC(0,1,10) -
VOLID(yyyy) NOVERIFY
```

where xxxx = the channel and unit address of the volume to be initialized.  
 yyyy = volume serial label of the disk volume

2. The following JCL describes initializing a volume mounted online to MVS.

```
//INITDISK EXEC PGM=ICKDSF
//SYSPRINT DD SYSOUT=A
//CUU DD DSN=SIPFILES,DISP=OLD,UNIT=xxxx,
VOL=SER=yyyy
//SYSIN DD *
INIT DDNAME(CUU),VTOC(0,1,10) VOLID(zzzz) -
NOVERIFY
```

where xxxx = device unit number which the volume is mounted on  
 yyyy = volume serial of the volume to be initialized  
 zzzz = volume serial label of the disk volume

3. The following JCL should be run to scratch any data sets on the volume.

```
//SCRATCH EXEC PGM=IEHPROGM
//DD0 DD UNIT=xxxx,VOL=SER=yyyy,DISP=OLD
//SYSIN DD *
SCRATCH VTOC,VOL=xxxx=yyyy,PURGE
/*
```

where xxxx = device unit number which the volume is mounted on  
 yyyy = volume serial of the volume to be scratched

4. Create the SIP output data sets. The values coded for SPACE are only guidelines.

**Notes:**

- xx = TPF release level. It must be the same as the value coded on the RELNN parameter of the GENSIP macro in stage I.
- yyyy = subsystem name. It must be the same as the value coded on the SSNAME parameter of the SSDEF macro in stage I or blank if SSDEF is not coded.
- zzzzz = volume serial number of an MVS volume.
- uuuu = unit.
- The block size for ACP.LINK.RELxx.yyyy must be understood in relation to the size of the FACE table to be generated. This relationship is:  

$$\text{BLKSIZE} * 512 \geq \text{the size of the largest FACE Table}$$

So, determine the size of the largest FACE table potentially to be used and divide by 512. The resulting number is the smallest BLKSIZE that can be used.

5. Compile and link-edit the FACE table generator (see "Compiling the FACE Table Generator" on page 188 for more information).



```

//ALLOC EXEC PGM=IEFBRI4
//DD1 DD DSN=ACP.OBJ.RELxx.yyyy,
//      SPACE=(TRK,(850,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD2 DD DSN=ACP.LINK.RELxx.yyyy,
//      SPACE=(TRK,(400,20)),
//      DISP=(,CATLG),DCB=(RECFM=U,LRECL=80,BLKSIZE=1200),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD3 DD DSN=ACP.SYMACR0.RELxx.yyyy,
//      SPACE=(TRK,(200,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD4 DD DSN=ACP.SYSRCE.RELxx.yyyy,
//      SPACE=(TRK,(150,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD5 DD DSN=ACP.SALTB.L.RELxx.yyyy,
//      SPACE=(TRK,(20,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=12,BLKSIZE=3504),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD6 DD DSN=TPFxx.ANTS.yyyy,
//      SPACE=(TRK,(20,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD7 DD DSN=ACP.CLIB.RELxx.yyyy,
//      SPACE=(TRK,(850,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD8 DD DSN=ACP.STUB.RELxx.yyyy,
//      SPACE=(TRK,(850,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD9 DD DSN=ACP.IMPORTS.RELxx.yyyy,
//      SPACE=(TRK,(75,15,25)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz

```

Figure 27. JCL for Allocating TPF Data Sets

```

//ALLOC EXEC PGM=IEFBRI4
//DD1 DD DSN=BDF.V1R1M3.BDFOBJ1.yyyy
//      SPACE=(TRK,(400,20)),
//      DISP=(,CATLG),DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz
//DD2 DD DSN=BDF.V1R1M3.BDFLNK1
//      SPACE=(TRK,(400,20)),
//      DISP=(,CATLG),DCB=(RECFM=U,LRECL=80,BLKSIZE=1200),
//      UNIT=uuuu,
//      VOL=SER=zzzzzz

```

Figure 28. JCL for Allocating TPFDF Data Sets

## Creating Program Tables

During stage I execution, SIP accesses member SPPGML of ACP.SIPGEN.RELv to determine which programs and versions must be assembled. This member already exists and contains all released programs including most of those created by SIP skeletons.

The program tables created by SIP using SPPGML are divided into separate groups:

- Dynamic link library (DLL) load modules
- Offline (OL) programs
- Object code only (OCO) programs
- Real-time (RT) programs
- Real-time special (RTS) programs
- User real-time (UR) programs
- C language real-time (CRT) programs (TARGET(TPF) only)
- C language user real-time (CUR) programs (TARGET(TPF) only)
- ISO-C assembler functions (ISA)
- ISO-C source segments (ISC)
- ISO-C load modules (ICL)
- Control program (CP) CSECTs
- Keypoints (KP).
- C++ language source code (CPP)

In addition to the program name, each entry may contain a version number, if applicable.

### Control Program (CP) Table

The CP table contains names and, where applicable, version numbers of all the CSECTs that comprise the control program. Each item in this table identifies a member which may consist of one or more copy cards for a specific CSECT. A copy card is an assembly control statement that contains a segment name. During program assembly, these statements will cause the appropriate segment to be assembled according to the sequence in which they appear. Copy cards are used when a CSECT is too large or complex to be called individually.

The user may not modify the CP table except to change a CSECT's version number when it is being replaced by a user-modified CSECT. In such cases, the user CSECT must be installed to the ACP.SRCE.CP.RELv library for that release before executing SIP stage II.

### Dynamic Link Library (DLL) Load Module Table

The DLL load module table contains all the load module names for DLLs with their associated version numbers. An entry exists in the ACP.LINK.RELv data set for each load module name. For each entry in the DLL list that IBM ships, there will be a build script entry in the ACP.CSRCE.RT.RELv data set.

### Keypoint (KP) Table

The KP table contains all keypoints, with version numbers, that must be in the system. These keypoints (except for CTK1, CTK2, CTK3, CTK4, and CTK9) initially exist as SIP skeletons in data set ACP.SIPGEN.RELv. A skeleton is an image of the actual unit enclosed in quotes in a punch statement and can have specific

values substituted. These skeletons are converted to source code and assembled during SIP stage II. Refer to each skeleton macro listing commentary for specific information on a particular SIP skeleton.

All names in the KP table, except general file keypoints, are 4 characters plus a 2-character version number. General file keypoints (CTKA and CTKV) are identified by general file version number GF.

The KP table must not be modified because of SIP's use of skeletons to create keypoint source code. However, any user keypoints can be added to TPF by placing them into the UR table.

### **Offline (OL) Table**

The OL table contains names and version numbers for all programs that will be executed in offline mode and are part of the released ACP.SRCE.OL.RELv and ACP.CSRCE.OL.RELv libraries. No changes can be made to the released OL table except to change program version numbers for user-modified programs or change the object code designation. In addition, the source code for all user-modified programs must be installed onto the ACP.SRCE.OL.RELv and ACP.CSRCE.OL.RELv libraries for that release before executing SIP stage II. Any user-developed offline programs must be assembled and link-edited without using SIP.

### **Real-Time, Real-Time Special, and User Real-Time Tables**

The real-time (RT) and real-time special (RTS) tables contain all system-controlled, ECB-dependent real-time program names with their associated version numbers, which are part of the released ACP.SRCE.RT.RELv and ACP.CSRCE.RT.RELv libraries. All program names are 6 characters long with the first 4 characters comprising the segment name, and the last 2 characters being the version number. Any user real-time program may be added to the user real-time (UR) table. User real-time programs will be processed in the same way as system real-time programs. If you want to replace a released program with your own version, just change the released version number to your own version number.

The entries for BAL programs that are called by DLMs are updated with STUB=YES, which means that the object name is added to the stub table (the ISB). During SIP stage II, a DLM stub is generated for each name in this list.

User programs will be accessed via GENSIP macro parameter USRCE, which identifies the user source library. The user source library must be a cataloged data set.

### **C Language Real-Time and User Real-Time Tables**

**Note:** The C language real-time (CRT) and user real-time (CUR) tables are for TARGET(TPF) only.

The C language real-time (CRT) and user real-time (CUR) tables contain all ECB-dependent C language real-time program names with their associated version numbers, which are part of the released ACP.CSRCE.RT.RELv libraries. All program names are 6 characters long, with the first 4 characters comprising the segment name and the last 2 characters the version number. Any C language user real-time program may be added to the CUR table. C language user real-time programs will be processed in the same way as system C language real-time programs. If you want to replace a released program with your own version, just change the released version number to your own version number.

The entries for the TARGET(TPF) programs that are called by DLMs are updated with STUB=YES, which means that the object name is added to the stub table (the ISB). During SIP stage II, a DLM stub is generated for each name in this list.

User programs will be accessed via GENSIP macro parameter USRCE, which identifies the user source library. The user source library must be a cataloged data set.

### ISO-C Assembler Function Table (ISA)

The ISA table contains all ISO-C assembler function names with their associated version numbers. An entry exists in the ACP.SRCE.RT.RELv data set for each ISO-C assembler function.

### ISO-C Source Table (ISC)

The ISC table contains all ISO-C source file names with their associated version numbers. An entry exists in the ACP.CSRCE.RT.RELv data set for each ISO-C source file.

### ISO-C Link Table (ICL)

The ICL table contains all the load module names (both library and application) with their associated version numbers. An entry exists in the ACP.LINK.RELv data set for each load module name. For each entry in the ICL table that IBM ships, there will be a build script entry in the ACP.CSRCE.RT.RELv data set. ICL entries for DLMs that are called by other DLMS have STUB=YES.

### C++ Language Source Table (CPP)

The CPP table contains all the C++ source file names with their associated version numbers. An entry exists in the ACP.CSRCE.RT.RELv data set for each C++ source file.

### Object Code Only (OCO) Table

The OCO table contains segments for which no source code is shipped. SIP will never attempt to assemble or compile a segment designated as OCO. Object code only programs are shipped as part of the TPF released ACP.OBJ.RELv library. All program names are 6 characters long, with the first 4 characters comprising the segment name and the last 2 characters the version number. Any object code only program may be added to the OCO table.

#### Notes:

1. CTKX, BXAX, COSY, CRS0, and CVZD (general file control record), all of which appear in the RTS table, must not be modified because of SIP skeletons, which are used to create the applicable source code.
2. The assembler language real-time programs are contained on 3 data sets. These data sets are as follows:

<b>ACP.SRCE.RT1.RELv</b>	All real-time programs that contain a letter A or B as the first character of the program name.
<b>ACP.SRCE.RT2.RELv</b>	All real-time programs containing the letter C as the first character of the program name.
<b>ACP.SRCE.RT3.RELv</b>	All other real-time segments.

3. The C/C++ language source files are contained on the following data sets:

<b>ACP.CSRCE.RT.RELv</b>	All C/C++ language source files. Build scripts and library interface scripts are also included.
<b>ACP.CHDR.RELv</b>	All C/C++ language headers.

4. The object code only programs are contained on the following data set:

## ACP.OBJ.RELv

All object code only programs.

**Note:** This ACP.OBJ.RELv data set will eventually contain all the object code needed to construct a TPF system. It is not intended to contain only OCO programs.

## Modifying the Program Tables

You may want to modify or add to any of the program tables, subject to the limitation specified for each table. If such a change is desired, update member SPPGML.

SPPGML consists of a series of SPPBLD macro calls. These macro calls consist of a list of program names, a library-type designator, a support function indicator, an indicator to define if the program is available in object code format, and a subsystem designator. The SPPGML macro is important in the SIP process because it provides the basis of selectability in the stage I assembly. A program module defined in SPPGML will be included or excluded from a user's system based on the options selected by the user in the stage I macros. The format of the SPPGML statement is as follows.

cc10      cc17

SPPBLD    TYPE,PGM1,PGM2,.....,PGMN,FUNC=yyy,OBJ=YES|NO,SS=ssname,OPT=n

Where:

### TYPE

This parameter defines the program library designation.

Valid library types are:

<b>DLL</b>	Dynamic link library
<b>RT</b>	Real-time program library
<b>OL</b>	Offline program library
<b>CP</b>	Control program library
<b>KP</b>	Keypoint program library
<b>RTS</b>	Real-time special program library
<b>UR</b>	User real-time program library
<b>CPP</b>	C++ language source code library
<b>CRT</b>	C language real-time program library (TARGET(TPF) only)
<b>CUR</b>	C language user real-time program library (TARGET(TPF) only)
<b>ICL</b>	ISO-C load module library
<b>ISA</b>	ISO-C assembler function library
<b>ISC</b>	ISO-C source library
<b>OCO</b>	Object code only program library.

**Note:** Library type RTS is used in the SPPGML and SPPBLD macros. This library type represents special repetitively used real-time programs processed by SIP. This library designation should not be used by user groups.

**PGM1–PGMn**

The program segment names to be associated with a particular library type.

**FUNC=yyy**

This parameter defines the Boolean switch usually set on or off by a SIP user macro. YYY specifies a name given that Boolean switch defined in the SIP GLOBAL definition macro, SPGLB. It is this switch that determines if a program or group of programs are included in the program tables. It is from the program tables that SIP gets the programs for assembly, link-edit, or execution. If a function switch is off (equal to 0), SPPBLD does not put the program names into the associated table. If the switch is on (equal to 1), the program names are installed into the associated program table.

**STUB=YES|NO**

This parameter is added to SPPBLD so that SIP can generate a DLM call stub for any program called by an ISO-C program. The default is NO.

**RENT=YES|NO**

This parameter is used to specify whether the source code for an ISC-type program is reentrant. The default is NO for type ISC. For type CPP, do one of the following:

- Code RENT=YES.
- Do not code the RENT parameter.

An error message occurs if you code RENT=NO for type CPP.

**DLL=YES|NO**

This parameter is used to specify whether the source code for type ISC is to become part of a dynamic link library (DLL). The default is NO for type ISC. For type CPP, do one of the following:

- Code DLL=YES.
- Do not code the DLL parameter.

An error message occurs if you code DLL=NO for type CPP.

**OBJ=YES|NO**

This parameter defines to SIP whether these C load modules are to be included in the link JCL created by SIP for the stage II process. If the OBJ parameter is coded YES, this informs SIP that the existing preassembled object version of the program is to be used in the link-edit or execution steps and assembly JCL is not required. If NO is specified, assembly JCL will be created in order to assemble the SOURCE module specified. The default value is NO.

**Notes:**

1. The OBJ parameter is ignored for OCO-type programs. OCO-type programs are never assembled or compiled.
2. See the coding of the ASMLL and OBJLIB parameters in “GENSIP” on page 247.

**SS=ssname**

This parameter defines to SIP the subsystem in which the programs are to be resident. This parameter is only valid in a TPF system containing the multiple database function (MDBF). See SSDEF. For library type CP, the default is BSS. For all other library types, the default is ALL.

**OPT=0|1|2**

This parameter specifies the optimization level of the program when the TYPE

parameter specifies ISC or CPP. The OPT parameter is ignored for all other types. If this parameter is not specified, the default is 2, which means optimization for C and C++ programs.

- 0** No optimization
- 1** OPT(1) for C programs; no optimization for C++ programs.
- 2** OPT(2) for C programs; optimization for C++ programs.

#### **LINK=YES|NO**

This parameter indicates whether SIP is to send link-edit JCL for a C load module.

- YES** Specifies that SIP is not to generate link-edit JCL in a stage II deck for this C load module. This parameter is ignored if you code LINKALL=YES on the GENSIP macro. ***This is the default.***
- NO** Specifies that SIP will generate link-edit JCL in a stage II deck for this C load module.

See “GENSIP” on page 247 for more information about the GENSIP macro.

Any of the previous library types can be modified in the SPPGML macro by using the MVS utility, IEBUPDTE, or any corresponding 80-column card image editor.

## **WTC Support**

SPIPML, released on library ACP.SIPGEN.RELv, is a list of the program components required by a WTC user in addition to those in SPPGML. SPIPML lists all the real-time programs required by the IPARS application. (Offline programs are listed separately in member SPWTC.) The comments made previously concerning modifications to the member names and inclusion of user versions apply equally to member SPIPML.

---

## **Creating a Site System Allocator**

The allocation process involves two sources of input: the IBMPAL deck and the user-defined program decks. The IBMPAL deck contains allocation statements required for TPF. The user-defined program decks contain the statements required for allocation of site-specific structures. The INDSN macro is provided, which allows up to 15 user decks to be specified.

**Note:** The IBMPAL deck should not be modified. Because there are fewer constraints on main storage than in previous TPF system releases, placement of statements early in the allocator deck is no longer required. This means that all user modifications to the allocations provided by IBM can be restricted to the user's own input decks.

## **Description of the Allocator Decks**

Creating the allocator decks is the first step in the program allocation process. They are ordered lists of real-time programs (including transfer vectors) and GFS pool record ID definitions. The output of SALO is the System Allocator Table (IDSSAL), which is used by the linkage editor (LEDT) to resolve VCONs for segments, and the IPATvv, the program allocation table containing the file and core addresses of E-type programs. No actual addresses are allocated by SALO; instead, allocation is done during system initialization (CTIN). (See the *TPF System Installation Support Reference* for more information about the system allocator.)



The segments listed in the IBMPAL macro have no implied order based on their position in the list. Core resident and file resident allocations may appear intermixed. The older notion of *core fast* has been dropped in favor of making all programs with a core resident requirement as much as possible remain in central memory at all times.

Included in the allocator deck are:

- Data-only segments
- RES0 applications
- BSS allocated programs
- User exits
- Segments allocated to all subsystems (except BSS)
- Segments allocated to all subsystems (including BSS).

See *TPF System Installation Support Reference* for more information.

---

## Record ID Attribute Table (RIAT)

The record ID attribute table (RIAT) contains information on every file type (both fixed and pool) in the database. This information includes the following pool file attributes:

- Size
- Duration
- Duplication.

The RIAT also includes the following information for both fixed and pool file records:

- Logging data
- Exception recording data
- Record restore
- User exit data
- VFA candidacy
- Locking attribute
- Record caching candidacy.

There is 1 entry in the RIAT for each record ID defined in the system.

## Creating the RIAT

RIATA macro calls, coded in the RIAT segment, create entries in the record ID attribute table (RIAT). This table contains information about all (fixed and pool file) IDs in the system. There must be 1 RIATA call for every ID in the database. It is used by VFA, logging, restoring, exception recording, GFS, the locking attribute, and record cache subsystem (RCS) support for processing options.

The RIAT is created by coding a deck of RIATA calls and assembling it. The RIAT deck consists of:

- A RIATA START call (see “RIATA START” on page 311)
- A series of RIATA ID= calls (see “RIATA ID” on page 305)
- A RIATA FINISH call (see “RIATA FINISH” on page 304).



You can code the RIATA calls in any order. When the RIAT is assembled, a RIAT is created. Error messages are generated for duplicate IDs or invalid parameter values.

A sample deck of RIATA calls is provided as the SPRIAT macro, which is a copy member in ACP.MACRO.RELxx. Modify these RIATA calls to fit each individual system (for example, only pool types coded in your RAMFIL statements should appear in your RIATA calls). Edit this deck and incorporate the applicable RIATA calls into the RIAT segment by either (1) including a COPY SPRIAT statement in the RIAT segment or (2) including the edited SPRIAT directly into the RIAT segment.

The IDs can be in character, hexadecimal, Boolean, or decimal format; the SPRIAT deck contains IDs in character and hexadecimal format. (A duplicate ID error message is generated for RIATA calls that code the same numeric ID in two different formats.)

To locate the RIAT, SIP stage I retrieves the data set name (coded in the INDSN macro) and the member name in which the RIAT resides from SPPGML. The RIAT is assembled as part of SIP stage II and is loaded into the system as part of the core image restart area. The RIATDSN= parameter of the INDSN SIP macro is used to indicate what RIAT segment to assemble and load. The default is the RIAT segment located in ACP.SRCE.OL.RELxx.

When the RIAT is assembled, a check is made to ensure that the pool types coded in RIATA calls are available (pool type availability is determined by input to the RAMFIL macro). If a RIATA call is coded with an unavailable pool type, a warning message appears in the assembly listing.

Positive feedback support allows you to understand what programs have been loaded onto your TPF system. It also provides information about changes that were made to the program allocation table (PAT) or RIAT; for instance, ordinal record 1 through the number of records that were defined will have entries added to them by positive feedback support for every program that is activated, deactivated, accepted, or excluded (that is, every time an operator enters the ZAPAT, ZAPGM, or ZRTDM MODIFY command). You must define #IPSFB 4-KB fixed file records to use positive feedback support.

## Accessing the RIAT Online

Although the attributes associated with a particular record ID are initialized when the RIAT is built, some of the attributes can be modified online.

The attributes that can be modified (using the ZRTDM command) include:

- Logging
- Exception recording
- Record restore
- User exit
- VFA candidacy
- Locking attribute
- Record caching candidacy.

**Note:** Attributes cannot be modified for IDs that are not defined in the RIAT, and table entries for IDs cannot be added after the RIAT is created. Pool

attributes cannot be changed with the ZRTDM command. In order to add pool attributes or new IDs, a new RIAT table must be created and loaded to the system.

Information in the RIAT is accessed by means of the RITID macro and the ZRTDM command.

---

## FACE Table Generation

The primary purpose of the FACE table generator (FCTBG) is to read in the definition of the TPF database, validate it, and produce a FACE table. The FACE table generator is a C program that runs offline under MVS control.

The source of input for the FACE table generator is the SIP stage I deck. The generated FACE table supports either FARF3/4 or FARF4/5 addresses. FARF6 addresses are also supported. The generator is run before SIP stage I. It creates a FACE table from existing SIP stage I decks defining FARF3 structures as well as from new decks defining FARF3, FARF4, FARF5, or FARF6 structures. Some SIP macro variables are put in a data set and included by SIP stage I so consistency checking can be done between RAMFIL and other SIP macros, additionally validating the FACE table when a SIP stage I deck is assembled.

The FACE table produced by FCTBG is in the form of an MVS object module. The MVS linkage editor is used to convert the object module to an MVS load module, which is then loaded by the TPF system. The MVS linkage editor is also used to attach user data to the FACE table.

## FACE Table Generator Input

Various statements, that follow, are included in SIP stage I decks. They are used by the generator to define the FACE table. These are the only statements and parameters read by the generator. When the stage I deck is updated with changes to any of these parameters, the generator must be run to create the corresponding FACE table. If any output from the FCTBG changes, the impact must be assessed on STCEQ, SYSEQC, DASD and format entries, and CYMZx SYCON macro elements. If any output from the FCTBG changes, the impact must be assessed on STCEQ, SYSEQC, DASD, IRCCR and format entries, and CYMZx SYCON macro elements. Because these are the only parameters affecting the FACE table if none of them change, the generator does not have to be run.

Macro	Comment	Reference
RAMFIL		"RAMFIL" on page 290
UFTFTI	If coded	"UFTFTI" on page 318
ONLFIL	DEVICEx, PERMx, EXTRx, DUPTYPx, NAMEDEVx parameters only	"ONLFIL" on page 282
SSDEF	SSNAME, SSUID1–4 parameters only	"SSDEF" on page 312
CONFIG	SYSID parameter only	"CONFIG" on page 219
RAM	HASHSZ and NFBACK parameters only	"RAM" on page 286

## FACE Table Generator Output

The product of the generator is:

<b>FCTBvv</b>	The FACE table as an object module.
<b>CYMZx</b>	The module configuration table as assembler source. This table will be copied in as part of SYCON.
<b>FMTRx</b>	Format cards used as input to the TPF formatter (FMTR).
<b>IRCCR</b>	The pool RCC conversion table (used by the GRRTC macro).
<b>STCEQ</b>	Record type names and equate values used by STC and DFAD.
<b>SYSEQC</b>	The record type equates for fixed file records as assembler source.
<b>SYFCTB</b>	A file containing validation information to be used during stage I processing.

#### **Standard output**

The report listing file.

## **Building a FACE Table**

The following is an outline for creating a FACE table for the first time.

1. If necessary, initialize the disk volume where the SIP data sets will reside, using ICKDSF.
2. Allocate the SIP output data sets.
3. Compile and link the FACE table generator, shipped as source (C code).
4. Code a SIP stage I deck.
5. Run the FACE table generator.
6. Assemble the SIP stage I deck.
7. Run SIP stage II.
8. Assemble CONK, CTSD, and IRCCR. Link-edit the FCTB with CONK, CTSD, IRCCR, and any user data as part of stage II.
9. Perform a full load.

Changing an existing FACE table is more complex because there are so many dependencies based on it. In addition to dependencies inherent in the product shipped by IBM, customer changes cause dependencies that IBM cannot account for. The following steps are sufficient for changing uncustomized FACE tables and associated data.

- Update the necessary input statements to the FACE table generator (RAMFIL, UFTFTI, and so on).
- Run the FACE table generator.
- Compare the formatter for the new database against the formatter for the current database. If the format has changed, the database will have to be reformatted and the appropriate loads performed. Database reorganization will be required.
- Determine what other outputs from the new FACE table generator run are different from the existing database outputs.
  - If the CYMZ40 entries have changed, segments that rely on this information will have to be reassembled and loaded.
  - If the SYSEQC equates have changed, any segments that rely on the changed equate s will have to be reassembled and reloaded.
  - If the SYFCTB file contents changed, rerun SIP stage I so that the proper validation can be performed.
- Assemble CONK, CTSD, and IRCCR. Link-edit the FCTB with CONK, CTSD, IRCCR, and any user data.

- Use GTSZ to determine the size of the new FCTB. If it will fit in the CIMR area (see CTKX source for the number of 4K ordinals reserved in the CIMR for the FCTB), only the FCTB needs to be loaded. If it does not fit, CTKX will have to be reassembled against the SKGTSZ produced from GTSZ, and it will also have to be loaded.

## Compiling the FACE Table Generator

The following is a sample JCL deck for compiling and link-editing the FACE table generator program. There are 4 steps:

1. Assemble SKDASD, which builds the DASD table source.
2. Assemble the DASD table source.
3. Compile all the segments that make up the FACE table generator.
4. Link-edit all the segments that make up the FACE table generator.

The TPF data set names have xx for version numbers and yyy for subsystem names; otherwise, the same data set names and specifications used during installation are used below.

**Note:** The pre-linkedit (PLKED) step of this JCL will have a return code of 4 because C library references cannot be resolved until the linkedit step. This warning should be ignored; it will not affect the FACE table generator.

```

/*
/* This job compiles the FACE Table Generator and builds the
/* DASDTBLE object deck. It then linkedits the FACE Table Generator
/* with the DASDTBLE object deck and then puts it into a link dsn
/*
/*
/* Assemble SKDASD which builds the DASDTBLE source
/*
/* note:
/* SKDASD source is in SIPGEN. Macros used are in MACRO
/* if your have macro hits used by SKDASD put the data set
/* containing your macros in front of .MACRO.STSIP40
/*
/*
//G1A1A EXEC PGM=ASMA90,COND=EVEN,REGION=3M,
//          PARM='DECK,XREF(SHORT),NOESD,NORLD'
//SYSUDUMP DD SYSOUT=A
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=1331),
//          OUTLIM=0,
//          SPACE=(TRK,(100,200),RLSE)
//SYSPUNCH DD DSN=&&DASDTMP;
//          ,UNIT=VIO,DISP=(MOD,PASS),
//          SPACE=(32000,(5,5))
//SYSLIN   DD DUMMY
//SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//          DD DSN=ACP.SIPGEN.RELxx,DISP=SHR
//          DD DSN=ACP.MACRO.RELxx,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1;,UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSIN    DD *
//          SKDASD
//          END
/*
/*
/* Assemble the DASDTBLE source
/*
//G1A1B EXEC PGM=ASMA90,REGION=3M,
//          PARM='DECK,NOOBJECT,XREF(SHORT)'
//SYSUDUMP DD SYSOUT=A
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=1331),
//          OUTLIM=0,

```

```

//          SPACE=(TRK,(100,200),RLSE)
//SYSPUNCH DD DSN=ACP.OBJ.RELxx.BSS(DASDTB),DISP=(OLD,KEEP)
//SYSLIN   DD DUMMY
//SYSLIB   DD DSN=SYS1.MACLIB,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1;,UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSIN    DD DSN=&&DASDTMP;,UNIT=VIO,DISP=(OLD,DELETE,DELETE)
//*
//*
//* Compile all the segments which make up the FACE Table Generator
//*
//CMPBD00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD01 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD02 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0240)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0240',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD03 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0340)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0340',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD04 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0440)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0440',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD05 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0540)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0540',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD06 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0640)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0640',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD07 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0740)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0740',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//              DD
//              DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD08 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0840)',

```

```

//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0840',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD09 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD0940)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD0940',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD10 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD1040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD1040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD11 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD1140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD1140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD12 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD1240)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD1240',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPBD13 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTBD1340)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTBD1340',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS01 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS02 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0240)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0240',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS03 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0340)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0340',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD

```

```

//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS04 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0440)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0440',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS05 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0540)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0540',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS06 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0640)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0640',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS07 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0740)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0740',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPPS08 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTPS0840)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTPS0840',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPVA00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTVA0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTVA0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPVA01 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTVA0140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTVA0140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPVA02 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTVA0240)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTVA0240',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPVA03 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTVA0340)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTVA0340',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG01 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',

```

```

//          MEMBER='FTRG0140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG02 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0240)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0240',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG03 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0340)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0340',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG04 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0440)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0440',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG05 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0540)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0540',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG06 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0640)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0640',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG07 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0740)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0740',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPRG08 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTRG0840)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTRG0840',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPGN00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTGN0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTGN0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPTD00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTTD0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTTD0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD

```



```

//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPTD01 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTTD0140)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTTD0140',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//CMPER00 EXEC EDCCLIB,INFILE='ACP.CSRCE.OL.RELxx.BSS(FTER0040)',
//          LIBRARY='ACP.OBJ.RELxx.BSS',
//          MEMBER='FTER0040',
//          CPARM='LONGNAME'
//COMPILE.SYSLIB DD
//          DD
//          DD DSN=ACP.CHDR.RELxx,DISP=SHR
//*
//* Linkedit all the segments which make up the FACE Table Generator
//*
//LKFBTGB EXEC PROC=EDCPL,LPARM='AMODE=31,RENT,MAP',
//  OUTFILE='ACP.LINK.RELxx.BSS(FCTBG40),DISP=OLD'
//SYSIN DD *
        INCLUDE OBJLIB(DASDTB)
        INCLUDE OBJLIB(FTGN0040)
        INCLUDE OBJLIB(FTBD0040)
        INCLUDE OBJLIB(FTBD0140)
        INCLUDE OBJLIB(FTBD0240)
        INCLUDE OBJLIB(FTBD0340)
        INCLUDE OBJLIB(FTBD0440)
        INCLUDE OBJLIB(FTBD0540)
        INCLUDE OBJLIB(FTBD0640)
        INCLUDE OBJLIB(FTBD0740)
        INCLUDE OBJLIB(FTBD0840)
        INCLUDE OBJLIB(FTBD0940)
        INCLUDE OBJLIB(FTBD1040)
        INCLUDE OBJLIB(FTBD1140)
        INCLUDE OBJLIB(FTBD1240)
        INCLUDE OBJLIB(FTBD1340)
        INCLUDE OBJLIB(FTER0040)
        INCLUDE OBJLIB(FTPS0040)
        INCLUDE OBJLIB(FTPS0140)
        INCLUDE OBJLIB(FTPS0240)
        INCLUDE OBJLIB(FTPS0340)
        INCLUDE OBJLIB(FTPS0440)
        INCLUDE OBJLIB(FTPS0540)
        INCLUDE OBJLIB(FTPS0640)
        INCLUDE OBJLIB(FTPS0740)
        INCLUDE OBJLIB(FTPS0840)
        INCLUDE OBJLIB(FTRG0140)
        INCLUDE OBJLIB(FTRG0240)
        INCLUDE OBJLIB(FTRG0340)
        INCLUDE OBJLIB(FTRG0440)
        INCLUDE OBJLIB(FTRG0540)
        INCLUDE OBJLIB(FTRG0640)
        INCLUDE OBJLIB(FTRG0740)
        INCLUDE OBJLIB(FTRG0840)
        INCLUDE OBJLIB(FTTD0040)
        INCLUDE OBJLIB(FTTD0140)
        INCLUDE OBJLIB(FTVA0040)
        INCLUDE OBJLIB(FTVA0140)
        INCLUDE OBJLIB(FTVA0240)
        INCLUDE OBJLIB(FTVA0340)
/*
//PLKED.OBJLIB DD DSN=ACP.OBJ.RELxx.BSS,DISP=SHR
//* END OF JOB

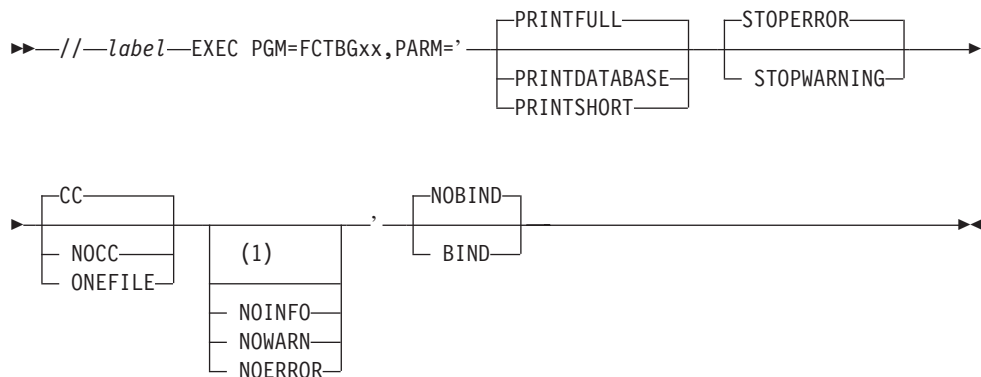
```

## FACE Table Generator Parameters

There are four groups of parameters for the generator. Two groups control the level of informative detail found in the report listing. Another group controls the acceptable level of errors after which the FACE table and other structures are no longer produced. The last group specifies the presence of carriage control in the report.

### FACE Table Generator Parameters

#### FACE Table Generator parameter syntax



#### Notes:

- 1 The default for this group of parameters is that messages of all severities are printed.

#### label

A symbolic name can be assigned to the EXEC invocation.

**xx** The version code associated with the FACE table generator.

#### PRINTFULL

This parameter specifies that all the input statements read in are written out in the report listing with appropriate messages attached to the statements. This is the default.

#### PRINTSHORT

This parameter specifies that only those input statements that have messages attached to them are written in the report listing with the messages attached to the statements.

#### PRINTDATABASE

This parameter specifies that only those input statements read by the FACE table generator are written in the report listing with appropriate messages attached to any statements.

#### STOPWARNING

This parameter stops the production of the FACE table (and other) output if there are warning messages, error messages, or fatal messages. The report listing continues to be produced.

**STOPERROR**

This parameter stops the production of the FACE table (and other) output if there are error or fatal severity-level messages. The report listing continues to be produced. This is the default.

**NOCC**

Specifying this parameter indicates that the output report should not include print control characters.

**CC**

Specifying this parameter indicates that the output report should include print control characters. This is the default.

**ONEFILE**

This parameter causes the output report to be combined with the status of the FCTBG run from OS/390.

**NOINFO**

This parameter indicates the report listing is not to contain informational messages.

**NOWARN**

This parameter indicates the report listing is not to contain warning messages or informational messages.

**NOERROR**

This parameter indicates the report listing must contain only fatal messages.

**NOBIND**

This parameter indicates that the linkage editor will be used to link-edit the FACE table. This is the default.

**BIND**

This parameter indicates that the binder will be used to link the FACE table. This parameter must be coded if the FCTB will exceed 16 MB in size.

## Running the FACE Table Generator

There are a number of JCL parameters required to run the generator. These should be checked to ensure their compatibility with the site. Note that *xx* and *yyy* are used again to represent the TPF version and subsystem name. The JCL is submitted for batch processing. The report listing file returns under the usual name of an MVS batch job.

Various C run-time parameters can be passed to the FACE table generator to control run-time storage usage.

```

/*
/* THIS PROC RUNS THE FACE TABLE GENERATOR
/*
/* &PVERS      - FACE TABLE GENERATOR VERSION TO RUN
/* &DATAOUT;    - DATASET WHICH WILL CONTAIN MACROS CREATED BY THE FACE TABLE GENERATOR
/* &DATAOUT1;-  - DATASET WHICH WILL CONTAIN ASSEMBLER DATA CREATED BY THE FCTBG
/* &FCTBOB     - DATASET WHICH WILL CONTAIN THE OBJECT FCTB DECK
/* &FVERS      - FACE TABLE OBJECT MODULE VERSION CODE
/* &FGNPARM    - FACE TABLE GENERATOR PARAMETERS
/* &IFILE      - INPUT DATASET
/* &FACE       - DATASET CONTAINING FACE TABLE GENERATOR
/* &WORKDA     - UNIT TYPE FOR WORKFILE
/* &WORKSPC    - SPACE ALLOCATED A WORK FILE
/* &WORKDCB    - DCB FOR LRECL OF 80
/* &OBDCB     - DCB FOR FCTB OBJECT DATASET
/* &SYSOUT1    - CONSOLE MESSAGES
/* &SYSOUT2    - SYSTEM PRINTER

```

```

/* &SYSOUT3 - FACE TABLE LISTING
/* &SYSOUT6 - SYSTEM DUMP FILE
/* &SYSOUT7 - C RUNTIME DUMP FILE
/*
//BLDFACE PROC PVERS=40,
//    FVERS=40,
//    FGPNARM='STOPERROR PRINTDATABASE NOCC NOBIND',
//    IFILE=,
//    FACE='ACP.LINK.RELxx.BSS',
//    FCTBOB='ACP.OBJ.RELxx.BSS',
//    DATAOUT='ACP.SYMACRO.RELxx.BSS',
//    DATAOUT1='ACP.SYSRCE.RELxx.BSS',
//    WORKDA='SYSDA',
//    WORKSPC='(32000,(30,30))',
//    TMPSPC='(TRK,(20,5))',
//    OBDCB='(RECFM=FB,LRECL=80,BLKSIZE=400)',
//    WORKDCB='(RECFM=FB,LRECL=80,BLKSIZE=3200)',
//    SYSOUT1='*',
//    SYSOUT2='*',
//    SYSOUT3='*',
//    SYSOUT6='*',
//    SYSOUT7='*'
/*
/* BUILD THE FACE TABLE
/*
//RUNFARF EXEC PGM=FCTBG&PVERS,PARM='&FGPNARM',REGION=1024K
//STEPLIB DD DSN=SYS1.EDC.SEDCLINK,DISP=SHR
//          DD DSN=SYS1.PLI.SIBMLINK,DISP=SHR
//          DD DSN=&FACE,DISP=SHR
//SYSOUT DD SYSOUT=&SYSOUT1
//SYSPRINT DD SYSOUT=&SYSOUT2
//SYSUDUMP DD SYSOUT=&SYSOUT6
//CEEDUMP DD SYSOUT=&SYSOUT7
//TMPFILE DD DSN=&TMPFILE,UNIT=&WORKDA,DISP=(NEW,DELETE,DELETE),
//          SPACE=&WORKSPC,DCB=&WORKDCB
//LSTFILE DD SYSOUT=&SYSOUT3
//ODECK DD DSN=&FCTBOB(FCTB&FVERS),DISP=OLD
//SIPFILE DD DSN=&TMPSPC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//STCFILE DD DSN=&TMPSTC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//IRCFILE DD DSN=&TMPIRC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//EQCFILE DD DSN=&TMPEQC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//FMATAFILE DD DSN=&TMPFMTA,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//FMTBFILE DD DSN=&TMPFMTB,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//FMTCFILE DD DSN=&TMPFMTC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//FMTDFILE DD DSN=&TMPFMTD,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//CYMZAFL DD DSN=&TMCYMZA,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//CYMZBFL DD DSN=&TMCYMZB,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//CYMZCFL DD DSN=&TMCYMZC,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//CYMZDFL DD DSN=&TMCYMZD,UNIT=&WORKDA,DISP=(MOD,PASS),
//          SPACE=&TMPSPC,DCB=&OBDCB
//INFILE DD DSN=&IFILE,DISP=SHR
/*
/*
/*
//COPYSIP EXEC PGM=IEBGENER,COND=(18,LE,RUNFARF)
//SYSPRINT DD SYSOUT=*

```

```

//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMP SIP,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(SYFCTB),DISP=OLD
//*
//COPYSTC    EXEC    PGM=IEBGENER,COND=(18,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPSTC,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(STCEQ),DISP=OLD
//*
//COPYIRC    EXEC    PGM=IEBGENER,COND=(18,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPIRC,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT1(IRCCR40),DISP=OLD
//*
//COPYEQC    EXEC    PGM=IEBGENER,COND=(18,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMP EQC,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(SYSEQC),DISP=OLD
//*
//COPY1      EXEC    PGM=IEBGENER,COND=(18,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPFMTA,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(FMTA),DISP=OLD
//*
//COPY2      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPFMTB,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(FMTB),DISP=OLD
//*
//COPY3      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPFMTC,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(FMTC),DISP=OLD
//*
//COPY4      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMPFMTD,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(FMTD),DISP=OLD
//*
//COPY5      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMCYMZA,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(CYMZA),DISP=OLD
//*
//COPY6      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMCYMZB,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(CYMZB),DISP=OLD
//*
//COPY7      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY
//SYSUT1     DD      DSN=&&TMCYMZC,DISP=(SHR,DELETE,DELETE)
//SYSUT2     DD      DSN=&DATAOUT(CYMZC),DISP=OLD
//*
//COPY8      EXEC    PGM=IEBGENER,COND=(8,LE,RUNFARF)
//SYSPRINT   DD      SYSOUT=*
//SYSIN      DD      DUMMY

```

```
//SYSUT1 DD DSN=&&TMCYMZD,DISP=(SHR,DELETE,DELETE)
//SYSUT2 DD DSN=&DATAOUT(CYMZD),DISP=OLD
//*
//      PEND
//*
//* End of PROC
//*
//BLDIT EXEC PROC=BLDFACE
//RUNFARF.SYSUDUMP DD DUMMY
//RUNFARF.INFILE DD *
```

## FACE Table Generator Error Messages

The error messages produced by the FACE table generator are listed in *Messages (System Error and Offline)* and *Messages (Online)* with FCTB as a message prefix along with explanations and responses.

### Structure of the Error Messages

The message format for the generator error messages is as follows:

```
RAMFIL TYPE=SSA,RECNO=01365,RECID=SPARE,DUPE=YES,BAND=1535
WARNING=>FCTB0026W: fn      ft      A1(build_ramfil):
=====>BAND IS NOT VALID FOR SPARE
```

The first line contains the source code that has the error. Messages that are not related to any particular line of source code are reported at the end of the job log. The second line contains the following:

```
severity=>FCTBxxxxy: source_file(function):
```

Where:

<b>severity</b>	is the severity of the problem
<b>xxxx</b>	is the error number
<b>y</b>	is also the severity
<b>source_file</b>	is the fully qualified name of the C source file issuing the message
<b>function</b>	is the function in that file issuing the message.

When messages associated with a fatal severity are found, processing ends after the message is issued. Messages with error level severities allow the generator to proceed, but only the report listing is produced. Messages with warning level severities also continue with validation and FACE table production, unless the STOPWARNING parameter is specified. Messages with an informational severity have no effect on program output.

## FACE Table Generator Report Listing

The report listing is divided into three sections:

1. The processing summary provides a count of the number of occurrences of stage I macros or statements found, along with relevant error information. The return code for the section appears at its end. One use of this list is to help ensure against misspellings of macro or statement names (such as, RAMF or RUMFIL), which the generator will not flag.
2. The contents of the source section varies depending on the parameters specified (NOWARN, PRINTSHORT, and so on). All the input read may be listed with all the informative notes, only the error input with the error messages, and so on. The format for the error messages in this section was described earlier.
3. The DASD map section describes the layout of disk storage that would result from the RAMFIL and other statements supplied in the stage I deck.

```
*****
*
*                                PROCESSING SUMMARY
* DESCRIPTION:    Below are statistics of the parsing process
*                and the highest error level.
*
*****
```

```
*****
*
* DATA BASE LISTING
*
```

```

* DESCRIPTION:   Below are data base related statements          *
*               including flagged statements.                     *
*               *                                               *
*****
SSDEF  TOTSS=4,          TOTAL NUMBER OF SUBSYSTEMS           X
      TOTSSU=8,          MAX NUMBER OF SSUSERS IN COMPLEX      X
      SSNAME=BSS,        BASIC SUBSYSTEM                       X
      BSSGEN=YES,        BASIC SUBSYSTEM                       X
      SSUID1=(HPN,HPN2)  TWO SSUSERS                           X
CONFIG APRNT=00E,        ADDRESS OF PRINTER FOR IPL SEQUENCE  X
      DUMPDEV=PRTR,      UPON IPL DUMPS ARE SENT TO PRINTER   X
      FQTK=NO,           NO FARE QUOTE TICKETING              X
      MAPSP=YES,         INCLUDE 3270 MAPPING SUPPORT          X
      MSGSW=YES,         INCLUDE MESSAGE SWITCHING SUPPORT     X
      USER=USA,          USER OF PARS                          X
      DCUSV=16,          ADDR. RANGE DASD CONTROL RANGE        X
      ENTERPRISE=DANBURY, X
      COMPLEX=TPFNET,     X
      SYSID=(B,C,D,E,Z,0), X
      PROC1=(20410,3090,,1), TPFXA1 LPAR                      X
      PROC2=(30410,3090,0,,0), TPF1 LPAR                      X
      PROC3=(40410,3090,,2,,3), TPF2 LPAR (ALSO 3090A/3090B) X
      PROC4=(50410,3090,4,,4), TPF3 LPAR                      X
      PROC5=(D0105,9221),   OLYMP processor                   X
      PROC6=(20215,9121),   KGN test floor processor           X
      PROC7=(20023,9021),   POK test floor processor           X
      VM=YES,              ACP WILL RUN UNDER VM/370           X
      RES=NO, DONT         INCLUDE RES APPLICATION             X
      TEST=NO,             WP TEST SYSTEM PARMETER            X
      ACF=YES,             ACF FEATURE                         X
      MPIF=YES,           MPIF FEATURE                         X
      CIPHR=(BOTH,ACP.OBJ.OC040), BOTH ENCRYPTION TECHNIQUES X
      WTOPCUNS=YES,        X
      VEQR=NO,            VIRTUAL = VIRTUAL                   X
      SELACT=YES,         Enable E-type Loader Sel. activate  X
      TPFDF=YES,          TPFDF EXCLUDED FROM SYSTEM          X
      TPFAR=YES,          X
      NEF=YES              @000.000
RAM    ECBNL=NOLIMIT,     NBR OF NESTING LEVELS FOR EACH ECB  X
      GFENS=60,           NUMBER OF GENERAL FILE SLOTS        X
      GFMOD=010,          SYMBOLIC MOD NBR - 1ST GENERAL FILE  X
      GSON=59,            MAX NBR OF SON GF STATUS TBL SLOTS   X
      NFBACK=2,           NUMBER OF KEYPOINT FALLBACK AREAS    X
      PHTBL=257,          NBR OF PRIMARY HOLD TABLE ENTRIES  X
      OHTBL=1001,         NBR OF OVERFLOW HOLD TABLE ENTRIES X
      OLDXPAT=1000,       XTRA PAT SLOTS                       X
      HASHSZ=0            NUM OF BYTES FOR HASH TABLE         X
ONLFIL DEVECA=(3390,3339), TYPE OF DEVICE                     X
      DUPTYPA=F,          DUPLICATION STATUS OF DEVICE         X
      PERMA=4,            NBR OF PERMANENTLY MOUNTED MODULES   X
      IPLABLE=2,          NBR OF IPLABLE MODULES               X
      VSNCHAR=SN,         ALPHA PORTION OF VSN                 X
      NAMDEVA=DEVA,       X
      VOLNOA=1,           STARTING VOLUME SERIAL NUMBER        X
      DEVECB=(3390,3339), TYPE OF DEVICE                     X
      DUPTYPB=F,          DUPLICATION STATUS OF DEVICE         X
      PERMB=4,            NBR OF PERMANENTLY MOUNTED MODULES   X
      NAMDEVB=DEVB,       X
      VOLNOB=5            STARTING VOLUME SERIAL NUMBER        X
UFTFTI STAGE=FARF45,MODE=FARF4, X
      UFTI=((0,08),(1,10),(2,07),(61,15),(62,13),(63,10), X
      (51,15),(52,13),(53,10),(10,08),(11,10),(12,07)), X
      UFTI6=((5,24),(2,10)) @000.000
RAMFIL TYPE=LSA,RECNO=00200,RECID=SPARE,DUPE=YES,BASE=00002 @000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0001, X
      UFTI4=(63,351),UFTI5=(53,351),EQU=441,USER=(*,C,1)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0002, X

```



```

        UFTI4=(63,352),UFTI5=(53,352),EQU=441,USER=(*,C,2)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0003,      X
        UFTI4=(63,353),UFTI5=(53,353),EQU=441,USER=(*,C,3)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0004,      X
        UFTI4=(63,354),UFTI5=(53,354),EQU=441,USER=(*,C,4)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0005,      X
        UFTI4=(63,355),UFTI5=(53,355),EQU=441,USER=(*,C,5)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0006,      X
        UFTI4=(63,356),UFTI5=(53,356),EQU=441,USER=(*,C,6)
.
.
.
RAMFIL TYPE=4SA,RECNO=000029,RECID=#BRIDT08,DUPE=YES,PRIOR=2,USER=HPN
RAMFIL TYPE=4SA,RECNO=000029,RECID=#BRIDT08,DUPE=YES,PRIOR=2,USER=HPN2
RAMFIL TYPE=4SA,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        UFTI6=((5,0),(5,2),(5,3)),BASE=220508,PSON=500 @000.000
RAMFIL TYPE=4SA,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=220603,PSON=740 @000.000
RAMFIL TYPE=4SA,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=220613,PSON=980 @000.000
RAMFIL TYPE=4SA,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=220708,PSON=1220 @000.000
.
.
.
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
        BASE=171713,PSON=496736 @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
        BASE=172113,PSON=498176 @000.000
RAMFIL TYPE=4SB,RECNO=105840,RECID=SPARE,DUPE=YES,BASE=172513 @000.000
RAMFIL TYPE=4SB,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=201913,PSON=27480 @000.000
RAMFIL TYPE=4SB,RECNO=000240,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=202008,PSON=27720 @000.000
RAMFIL TYPE=4SB,RECNO=029880,RECID=POOL6,DUPE=YES,POLID=LT,      X
        BASE=202103,PSON=27960 @000.000
.
.
.
RAMFIL TYPE=4SB,RECNO=430784,RECID=SPARE,DUPE=YES,BASE=213908 @000.000

```

```

*****
*
* MISCELLANEOUS RAMFIL INFORMATION
* DESCRIPTION: Below is POOL type information and other
* miscellaneous RAMFIL information
*
*****

```

```

THE FOLLOWING POOL TYPES WERE DEFINED FOR DEVICE TYPE A
small short term non-duplicated
small long term duplicated
large short term non-duplicated
large long term duplicated
4K short term non-duplicated
4K long term duplicated
4K long term duplicated FARF6
THE FOLLOWING POOL TYPES WERE NOT DEFINED FOR DEVICE TYPE A
small long term non-duplicated
small long term duplicated FARF6
large long term non-duplicated
large long term duplicated FARF6
4K long term non-duplicated
THE FOLLOWING POOL TYPES WERE DEFINED FOR DEVICE TYPE B
4K short term non-duplicated
4K long term duplicated

```

```

4K long term duplicated FARF6
THE FOLLOWING POOL TYPES WERE NOT DEFINED FOR DEVICE TYPE B
small long term non-duplicated
small short term non-duplicated
small long term duplicated
small long term duplicated FARF6
large long term non-duplicated
large short term non-duplicated
large long term duplicated
large long term duplicated FARF6
4K long term non-duplicated
THERE ARE 6503 ENTRIES IN THE HASH TABLE

```

```

*****
*
*                               DASD MAP, RESULTS OF FCTB GENERATION
*
* DESCRIPTION:  Below is the DATA BASE layout based on the Stage I*
*               input. All addresses are in hexadecimal. All
*               other numbers are in decimal. Descriptor for
*               FARF3 is the band number and for FARF4 or FARF5
*               is the UFT/FTI.
*
*
*****

```

```

+++++
Rec Type      Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV
=====
SPARE         L    Y      200  0000 0000 0002 01  0001 0000 0005 01  A
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0000 0000 0005 02  0000 0000 0005 04  A  N  *  C 1  HPN  C 1  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD5F0001  FD5F0011  0  4  63/351
          F5 @  D55F0000  D55F0004  0  4  53/351
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0001 0000 0005 04  0001 0000 0005 06  A  N  *  C 2  HPN  C 2  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD600001  FD600011  0  4  63/352
          F5 @  D5600000  D5600004  0  4  53/352
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0000 0000 0005 07  0000 0000 0005 09  A  N  *  C 3  HPN  C 3  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD610001  FD610011  0  4  63/353
          F5 @  D5610000  D5610004  0  4  53/353
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0001 0000 0005 09  0001 0000 0005 0B  A  N  *  C 4  HPN  C 4  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD620001  FD620011  0  4  63/354
          F5 @  D5620000  D5620004  0  4  53/354
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0000 0000 0005 0C  0000 0000 0005 0E  A  N  *  C 5  HPN  C 5  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD630001  FD630011  0  4  63/355
          F5 @  D5630000  D5630004  0  4  53/355
+++++

Rec Type      Prior  Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  Users:  Owner:  In Res
=====
#QS0JS        1      L    Y      5  0001 0000 0005 0E  0001 0000 0005 10  A  N  *  C 6  HPN  C 6  Y  Y
          Beginning FARF@  Ending FARF@  Beg Ord  End Ord  Descriptor
          -----
          F4 @  FD640001  FD640011  0  4  63/356
          F5 @  D5640000  D5640004  0  4  53/356
+++++

```

```

+++++
Users:
Rec Type      Prior  Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  SSU  P  I
=====
#BRIDT08      2      4      Y      29  0001 089A 0003 08  0001 089A 0004 0A  A  HPN  *  *
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF4 @      F4073991  F4073A01  100  128  61/231
      FARF5 @      CC073864  CC073880  100  128  51/231
+++++

Rec Type      Prior  Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  SSU  P  I
=====
#BRIDT08      2      4      Y      29  0000 089A 0004 0B  0000 089A 0006 01  A  HPN2 *  *
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF4 @      F40A0191  F40A0201  100  128  61/320
      FARF5 @      CC0A0064  CC0A0080  100  128  51/320
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      500      4      Y      240  0000 089D 0008 01  0001 089E 0002 0C  A  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000001F4  00000500000002E3  500  739  5/0
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      740      4      Y      240  0000 089E 0003 01  0001 089E 000C 0C  A  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000002E4  00000500000003D3  740  979  5/0
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      980      4      Y      240  0000 089E 000D 01  0001 089F 0007 0C  A  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000003D4  00000500000004C3  980  1219  5/0
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      1220     4      Y      240  0000 089F 0008 01  0001 08A0 0002 0C  A  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000004C4  00000500000005B3  1220  1459  5/0
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL      496736     4      Y      1440  0000 06B5 000D 01  0001 06B9 000C 0C  B  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF4 @      086F5181  086F67FD  496736  498175  2/13
      FARF5 @      30229460  302299FF  496736  498175  12/4
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL      498176     4      Y      1440  0000 06B9 000D 01  0001 06BD 000C 0C  B  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF4 @      086F6801  086F7E7D  498176  499615  2/13
      FARF5 @      30229A00  30229F9F  498176  499615  12/4
+++++

Rec Type      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV
=====
SPARE      4      Y      105840  0000 06BD 000D 01  0001 07E3 000C 0C  B
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      27480     4      Y      240  0000 07E3 000D 01  0001 07E4 0007 0C  B  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000006B5  00000500000006C47  27480  27719  5/0
+++++

Rec Type      PS0N      Size  Dupe  Rec Num  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
POOL6      27720     4      Y      240  0000 07E4 0008 01  0001 07E5 0002 0C  B  N  LT  0  2
      Beginning FARF@  Ending FARF@  Beg  Ord  End  Ord  Descriptor
      FARF6 @      00000500000006C48  00000500000006D37  27720  27959  5/0
+++++

```

```

+++++
Rec Type   PSON              Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV  DUMP  POLID  LOMOD  #MODS
=====
      POOL6              27960    4    Y      29880  0000 07E5 0003 01  0001 0838 0002 0C  B   N   LT    0    2
              Beginning FARF0  Ending FARF0  Beg Ord  End Ord  Descriptor
      FARF6 @  0000050000006D38  000005000000E1EF  27960  57839  5/0
+++++
Rec Type   Size  Dupe  Rec Numb  Beginning MCHR  Ending MCHR  DEV
=====
SPARE      4    Y      430784  0000 085B 0008 01  0001 0D08 0002 04  B

```

## Assembling CONK, CTSD, and IRCCR

The JCL for assembling CONK, CTSD, and IRCCR, and link-editing them with the FACE table (FCTB) follows. This JCL is part of stage II and is included here only for convenience.

```

//LINKFCTB JOB MSGLEVEL=1,CLASS=S
//I3B6L EXEC PGM=ASMA90,REGION=3072K,TIME=20,
// COND=EVEN,PARM='DECK,NOOBJECT,XREF(SHORT)'
//SYSUDUMP DD SYSOUT=A
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=1331),
// OUTLIM=0,
// SPACE=(TRK,(100,200),REL)
//SYSPUNCH DD DSN=ACP.OBJ.REL40.BSS(CONK40),DISP=OLD
//SYSLIN DD DUMMY
//SYSLIB DD DSN=ACP.SYMACRO.REL40.BSS,DISP=SHR
// DD DSN=ACP.MACRO.REL40,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSIN DD DSN=ACP.SRCE.OL.REL40(CONK40),
// DISP=SHR
/*
//I3B7L EXEC PGM=ASMA90,REGION=3072K,TIME=20,
// COND=EVEN,PARM='DECK,NOOBJECT,XREF(SHORT)'
//SYSUDUMP DD SYSOUT=A
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=1331),
// OUTLIM=0,
// SPACE=(TRK,(100,200),REL)
//SYSPUNCH DD DSN=ACP.OBJ.REL40.BSS(IRCCR40),DISP=OLD
//SYSLIN DD DUMMY
//SYSLIB DD DSN=ACP.SYMACRO.REL40.BSS,DISP=SHR
// DD DSN=ACP.MACRO.REL40,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(1700,(400,50))
//SYSIN DD DSN=ACP.SRCE.OL.REL40(IRCCR40),
// DISP=SHR
/*
//I3B8L EXEC PGM=IEWLF440,REGION=3072K,
// PARM='LET,LIST,XREF,DCBS,SIZE=(512K,64K),NCAL'
//SYSPRINT DD SYSOUT=A
//SYSLMOD DD DSN=ACP.LINK.REL40.BSS,DISP=OLD,DCB=BLKSIZE=1200
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(7294,(100,100))
//OBJLIB DD DSN=ACP.OBJ.REL40.BSS,DISP=SHR
//SYSLIN DD *
INCLUDE OBJLIB(FCTB40)
INCLUDE OBJLIB(CTSD40)
INCLUDE OBJLIB(CONK40)
INCLUDE OBJLIB(IRCCR40)
NAME FCTB40(R)
/*

```



---

## Stage I: Coding the SIP Macros

This section provides details for specifying each SIP macro. The macros are listed in alphabetical order and their function and parameters are summarized at the beginning of each page. Additional details and comments on each parameter are described after each summary. All default values are indicated when they are included within the parameter.

---

### Coding TPF with the Multiple Database Function

- When generating a subsystem that is **not** the basic subsystem, you must concatenate as part of the SYSLIB DD statements, the data SYMACRO data set created as part of the SIP run that created the basic subsystem. This is to provide the stage I run with the LOOK BACK macro BSSSET. This macro defines to the non-basic subsystem the support features generated in the basic subsystem.
- The SSDEF macro must be coded to define the MDBF feature.
- Code all macros and parameters unless specifically told not to do so.
- When coding the SIP macros for a subsystem that is **not** the basic subsystem, use the basic subsystem SIP input as a base. Change only the items required for the particular subsystem. Remember to include the BRELN parameter in the GENSIP macro.
- Thoroughly review the SIP report generated as a result of executing the SIP input. This could save you having to completely redo the system generation process after you have completed all the assemblies.

## BBSAT

The BBSAT macro must be coded if there are any BSC multipoint lines directly connected to the system being generated. Otherwise it should be omitted.

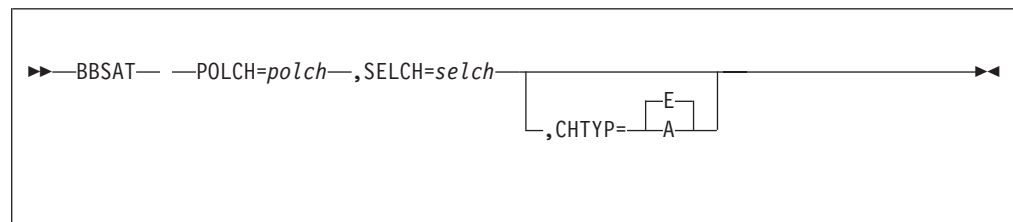
The BBSAT macro is used to specify valid sets of polling and selection characters in the BSC multipoint network. They will eventually be related to specific lines in the SCK and 3745 PEP generations.

One BBSAT macro should be coded for each unique set of poll and selection characters in the network. A maximum of 64 BBSAT macros may be coded. The minimum number of BBSAT macros is equal to the maximum number of drops on any one BSC multipoint line. This is because each drop on a BSC multipoint line must be assigned a unique poll and selection sequence (i.e., one BBSAT macro). There is no requirement, however, for uniqueness of poll and selection characters across BSC multipoint lines.

This macro is used to create the BSC station address table (BSAT, segment CRS0).

The BBSAT macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



### **POLCH=polch**

The poll address characters to be used for a station on a multipoint BSC line. There may be from 1 to 7 poll characters. They are coded as hexadecimal numbers and represent a unique station poll address.

### **SELCH=selch**

The selection address characters to be used for a station on a multipoint BSC line. There may be from 1 to seven select characters. They are coded as hexadecimal numbers and represent a unique station poll address.

### **CHTYP**

The type of poll and select characters specified.

**E** If coded as E or defaulted to E, the characters are EBCDIC.

**A** If coded as A, the characters are UASCII characters.

**Note:** The poll and selection characters must be the characters for the same station on the same BSC line.

If the characters are EBCDIC, then the following hexadecimal characters are not valid:

01, 02, 03, 10, 1F, 26, 2D, 32, 37, 3D



If the characters are UASCII, then the following hexadecimal characters are not valid:

01, 02, 03, 04, 05, 10, 15, 16, 17, 1F

For a tributary station on a BSC multipoint line, the first character of the poll and selection addresses must be the same except for bit 2 of the Hex byte. Bit 2 must be 0 in the poll character and 1 in the selection character. In addition, for a tributary station, a TADDR parameter in the 3745P EP generation must be consistent with one of the BBSAT macro definitions.

## Examples

**Example 1:** A BSC multipoint station having a poll address character of *B* in EBCDIC and a selection address character of X'E1' would be coded as follows:

```
BBSAT POLCH=C2,SELCH=E1
```

**Example 2:** A BSC multipoint station having poll address characters of *A* and *B* in ASCII format and selection address characters of *a* and *b*, also in ASCII, would be coded as follows:

```
BBSAT POLCH=4142,SELCH=6162,CHTYP=A
```

## References

None.

## BSNCT

The BSNCT macro must be coded if there are any BSC lines associated with the system being generated.

This macro is used to create the BSC Station Name Conversion Table (SNCT, segments CRS1-CRSG).

## Format

```

▶▶—BSNCT— —STANM=stanm—,CPUID=procid—,SYMLN=symln—,STANO=stano————→
▶—,APPLN=appln————→◀◀

```

### STANM=*stanm*

This parameter is the station name. It must be four characters. The same station name may appear in up to 16 BSNCT macros, indicating multiple paths to the same logic endpoint. However, if they do contain the same station name, each of those BSNCT macros must have a different symbolic line number (SYMLN parameter) and the same application name (APPLN parameter). A logical endpoint in the TPF network may be defined by this parameter in a BSNCT macro or by the APLIC parameter in a MSGRTA macro, but not by both.

**Note:** For speed in processing the BSNCT macros should be in collating sequence by station name. If not in collating order, SIP will sort the input but this will slow down the SIP Stage I process.

### CPUID=*cpuid*

This parameter is the ID of the host processor to which this station is attached. It must be entered as a single character in the range of A–Z or 0–9. This parameter, the APROC parameter of the MSGRTA macro, and the SYSID parameter of the CONFIG macro all refer to the same unique set of characters for processor identification across the network and should be coded with consistency across the network.

### SYMLN=*symln*

This parameter is the symbolic line number of the BSC line to which the station is attached. The number is hexadecimal and can be one or two characters long. The LINES macro contains a description of symbolic line number assignments. This is a required parameter for BSNCT macros describing BSC stations directly connected to the system being generated. (It may also be coded, for documentation purposes, in BSNCT macros describing BSC stations connected to a remote processor, though it is not required.)

### STANO=*stano*

This parameter is the station number. It is the relative position of the station on the line. The number is hexadecimal with the minimum value = 0 and the maximum value = 3F (63 decimal). There must be one and only one BSNCT macro for each unique combination of CPUID/SYMLN/STANO.

### APPLN=*appln*

This parameter is the name of the application to which messages received from

this station are to be routed. It must be four characters long. This application name must also be defined by a MSGRTA macro. The same APPLN may appear in multiple BSNCT macros. This may occur either because the same station name (STANM) appears in multiple BSNCT macros, or because multiple stations (different STANM parameters) messages are being routed to the same application.

**Note:** For a tributary station on a BSC multipoint line, only one BSNCT macro should be coded per line. This one BSNCT should contain the CPUID of the system being generated, a station number of 0, and a logical application name (that is, one which is defined in a MSGRTA statement as being resident in the system being generated).

A maximum of 16 paths to the same STANM are allowed. A maximum of 1300 BSNCT macros are allowed.

## Examples

An application named TEST is assigned to station TT00. This station is attached to a processor which has an ID of B. The symbolic line number associated with this BSC station is 26 (DEC. 38) and this station is the first station on the line. Note that the line numbers are determined by the LINES macro. A BSNCT statement for the above description would be coded as follows:

```
BSNCT STANM=TT00,CPUID=B,SYMLN=26,STAN0=00,APPLN=TEST
```

## References

None.

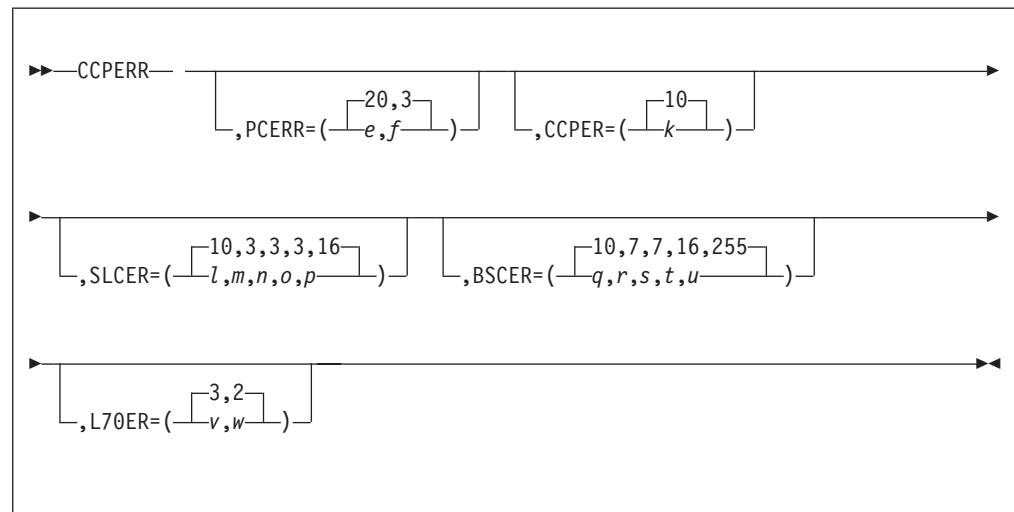
## CCPERR

The CCPERR macro is used to specify values for initiating the error recovery procedure by the communication control program (CCP) for non-SNA lines. When error counts reach the specified values, line shutdown is started. That is, the line is invalidated in the line status table (LSTB). Before coding this macro, it is recommended that you read about the communications control program in *TPF Non-SNA Data Communications Reference*.

This macro need not be coded if the default values shown below are acceptable. SIP automatically assigns these values if the macro is omitted, although a warning message is issued (GEN151). These values are stored in the LINEQC and SYSETK macros.

The CCPERR macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



**Note:** All values entered must be greater than zero.

**PCERR=(20,3)|(e,f)**

Specifies errors for the primary CRAS as follows:

- e* Maximum number of line errors.
- f* Maximum number of retries.

**CCPER=(10)|k**

Specifies interval (minutes) to update CCP error counts and search for lost interrupts.

**SLCER=(10,3,3,3,16)|(l,m,n,o,p)**

Specifies synchronous link (SLC) error counts as follows:

- l* Maximum control unit error count for SLC lines.
- m* Output unit check maximum retry count for SLC lines.
- n* Input unit check maximum retry count for SLC lines.
- o* Output unit check maximum error count for SLC lines.

$p$  Input unit check maximum error count for SLC lines.

**BSCER=(10,7,7,16,255)|(q,r,s,t,u,v)**

Specifies bisynchronous line (BSC) error counts as follows:

$q$  Maximum BSC control unit error count.

$r$  Maximum BSC output retry count.

$s$  Maximum BSC input retry count.

$t$  Maximum BSC output line error count.

$u$  Maximum BSC input line error count.

**L70ER=(3,2)|(v,w)**

Specifies error counts for 3270 Local as follows:

$v$  3270 Local error retry value. This specifies the number of times a write operation is retried after an I/O error. If omitted and 3270 local exists in the system, a default value of 3 will be substituted. The minimum value that can be coded is 3 and the maximum value is 10.

$w$  3270 Local restart cycle value. This specifies the number of times that an attempt is made to restart a control unit after an uncorrectable I/O error. If omitted and 3270 local exists in the system, a default value of 2 will be substituted. The minimum value that can be coded is 2 and the maximum value is 10.

## Examples

None.

## References

None.

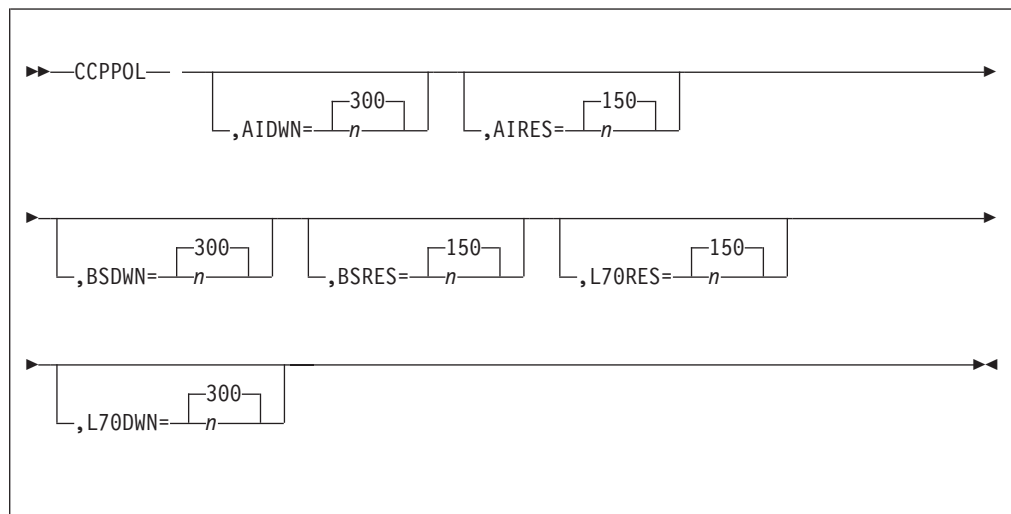
## CCPPOL

The CCPPOL macro is used to specify shutdown and restart levels for polling each non-SNA line type based on the size of the input list. Polling will stop when the size of the input list exceeds the shutdown level for that line type. Polling will restart when the size of the input list is less than the restart level. For SLC, these values are used to control the message rate. If the shutdown level is exceeded, a STOP ALL is sent and no data is sent across the link until the input list is below the restart level. Before coding this macro, it is recommended that you read about the communications control program in *TPF Non-SNA Data Communications Reference*.

This macro need not be coded if the default values are acceptable. SIP automatically assigns these values if the macro is omitted although a warning message is issued. These values are stored in keypoint A (CTKA).

The CCPPOL macro is coded when generating a 'base only' system or when generating the basic subsystem in a multiple database function (MDBF) environment.

### Format



**AIDWN=300|n**

Synchronous link (SLC) line shutdown level.

**AIRES=150|n**

SLC line restart level.

**BSDWN=300|n**

Bisync line shutdown level.

**BSRES=150|n**

Bisync line restart level.

**L70RES=150|n**

3270 local restart level.

**L70DWN=300|n**

3270 Local shutdown level.

**Note:** Default values provided have been found to be sufficient in running test systems and are provided as possible starting points for production systems. All CCPPOL parameters may be omitted.

## **Examples**

None.

## **References**

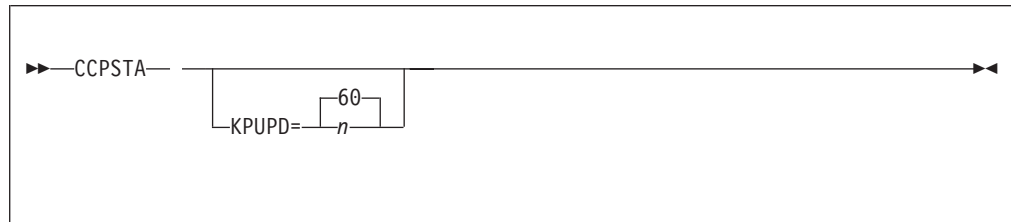
None.

## CCPSTA

The CCPSTA macro is used to set the CCP keypoint update value. This macro need not be coded if all of the default values shown are acceptable, although a warning message will be issued. SIP will automatically assign these values if the macro is omitted.

The CCPSTA macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



**KPUPD=60|n**

Communication control program (CCP) keypoint update value (decimal) in seconds. This value controls the time interval between keypoint updating. Any value greater than zero can be used to suit the user's requirements.

## Examples

None.

## References

None.



## CLOCKS

The CLOCKS macro is used to specify the current status of the clocks used in the online system. The parameters are used to set up the time fields in CTKA, CTKB and CTKX.

The CLOCKS macro is required.

## Format

```

▶▶—CLOCKS— —GDIF=shhmm—,LST=hhmm—,DATE=mm/dd/yy—,DELTIM=hhmm————▶
▶—,SDIF=shhmm————▶▶

```

### **GDIF=*shhmm***

Specifies difference between Local Standard Time and Greenwich Mean Time (GMT–LST) in signed (+,–) EBCDIC 2400 hour representation. The difference must be between –1200 and +1200.

### **LST=*hhmm***

Specifies local time in EBCDIC 2400 hour representation to be used before the system is cycled above 1052 state for the first time.

### **DATE=*mm/dd/yy***

Specifies the date the system clocks will use before the system is cycled above 1052 state for the first time, where *mm*=month, *dd*=day, and *yy*=year. If the year (*yy*) is between 73 and 99 (that is, between the years 1973 and 1999), assume the date is 19xx; otherwise, assume the year is 2000 or above (2000–2072).

**Note:** Before the system is cycled above 1052 state for the first time, the perpetual minutes clock and last midnight values in CTKB are initialized using the values specified by the DATE= and LST= parameters. Time and date displays will reflect these values. When the system is cycled above 1052 state, the clock values in CTKB are replaced with values calculated using a base date of 01/03/66 and the current time from the hardware TOD clock. Time and date displays will then reflect the current time and date.

### **DELTIM=*hhmm***

Specifies the subsystem time difference from the system clock. This parameter is specified as hh (hours) and mm (minutes). A maximum of up to 24 hours may be specified. This parameter is only valid in a TPF system with multiple database function (MDBF) support and should only be specified for nonbasic subsystem type generations.

### **SDIF**

Specifies the difference between Greenwich Mean Time (GMT) and the system clock (GMT–system clock) in signed (+,–) EBCDIC 2400 hour representation. The difference must be between –1200 and +1200. The default value is 0.

## CLOCKS

### Examples

The following specifies a plus 5-hour difference between the local time of the installation and Greenwich Mean Time. The local time is 3:00 P.M. and the date is May 21, 1996. Because SDIF is not specified the system clock is assumed to be set to GMT. SDIF is set to 0 by default.

```
CLOCKS GDIF=+0500,LST=1500,DATE=05/21/96
```

### References

None.

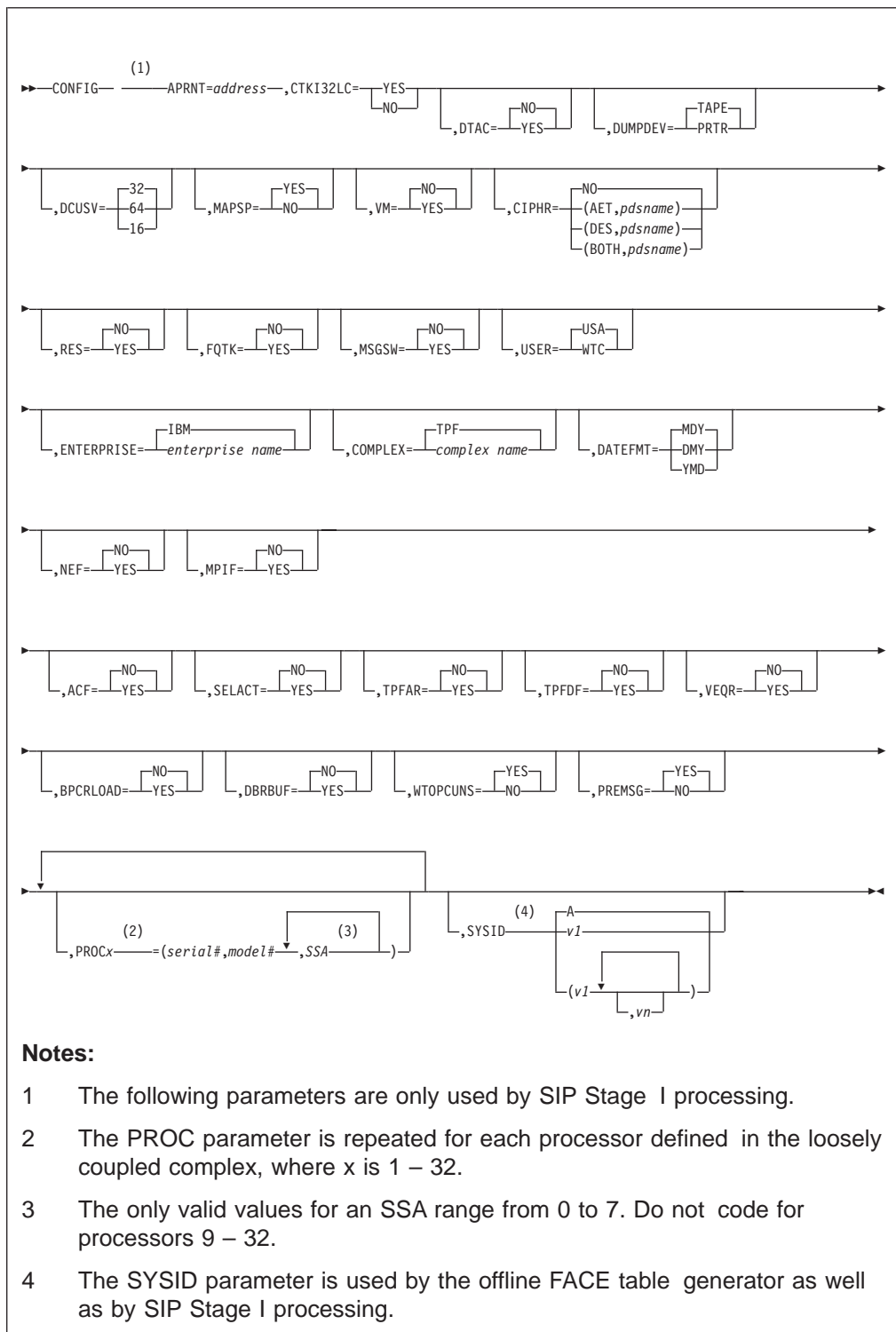
---

## CONFIG

The CONFIG macro is used to define some of the hardware and software configuration specifications for the installation.

The CONFIG macro is required.

## CONFIG Format



### APRNT=address

Three-digit physical address, in hexadecimal of a printer device to be used by the IPL program and the in-core dump formatter program for printing dumps. The first digit must be 0.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### CTKI32LC

Specifies whether keypoint I (CTKI) should be in 32-way loosely coupled format.

- YES** CTKI is generated with the 32-way loosely coupled processor support flag (IC032LC) set on, indicating that CTKI is in 32-way loosely coupled format. YES must be coded if one of the following is true:
- This is an existing loosely coupled complex and the ZMIGR command with the CTKI and CONVERT parameters specified has been completed successfully.
  - This is a new complex with 32-way loosely coupled processor support on all processors.
- NO** CTKI is generated with the 32-way loosely coupled processor support flag (IC032LC) set off, indicating that CTKI is not in 32-way loosely coupled format. NO must be coded if one of the following is true:
- This is an existing loosely coupled complex and the ZMIGR command with the CTKI and CONVERT parameters specified has not been completed successfully.
  - The ZMIGR command with the CTKI and FALLBACK parameters specified has been completed and CTKI is not in 32-way loosely coupled format.

### DTAC

This parameter limits the amount of data presented to the dump data user exit.

- NO** Detached data blocks will not be included.
- YES** The number of data blocks presented to the user exit will be limited by the constant DETDATA in CZOCP.

### DUMPDEV

Specifies the device to which systems error dumps are to go.

- TAPE** All system error dumps will go to the real time tape (RTA or RTL).
- PRTR** Program ICDF will be loaded as part of the in-core online system and all system error dumps will be printed online, on the printer specified in the APRNT parameter. The PRTR option should only be specified for test systems, because all processing is suspended while the dump is being printed.

It is recommended that you dump to tape on a production system, because it is faster than going to a printer.

### DCUSV=32|64|16

Specifies the range of installed addresses on the DASD control unit interface cards. Valid values are 16, 32 or 64.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

**Note:** If a 3880 storage director is set to answer to address range A0–AF, the DCUSV value will be 16. If it answers to address range A0–BF, then the DCUSV value will be 32. The value 64 is only valid for a 3990 storage director. The current locking architecture may not support 64 DASD behind a storage director.

## CONFIG

### MAPSP=YES|NO

Specifies the inclusion of 3270 mapping support.

See *TPF Data Communications Services Reference* for more information about mapping support. If MAPSP=YES is coded, and the system being generated is a base only system or the basic subsystem in an MDBF environment, then 3270 support must also be included by coding the MAXRVT parameter of the SNAKEY macro as non-zero at network definition time.

### VM=NO|YES

Two options in keypoint record A (CTKA) are generated. The *CP in test mode* bit is used by the command editor to allow or disallow the use of the ZTEST command. The *tapes in test mode* bit is used exclusively to control the use of general tapes. Note that PTV also uses these options. These options are also set by specifying PTV to SIP.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

**Note:** Either option can run native, or under VM. A system generated with the VM=NO options, when actually running under VM, is aware that it is running under VM. Conversely, a system generated with the VM=YES option is capable of running native.

Loosely coupled systems may not be run under VM. Specify VM=NO (or allow it to default) for all loosely coupled systems.

### CIPHR=(NO|AET|DES|BOTH,*pdsname*)

Specifies whether data encryption support is to be included in the system and the type of encryption technique desired.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

**NO** No data encryption support provided.

**AET** Provides support for the original IBM 3614 encryption technique. This technique will now be referred to as the Alternate Encryption Technique (AET). Specify the AET parameter with the name of a partitioned data set (PDS) that contains the BQKCIPH module or SIP processing will not be completed.

**DES** Provides support for the data encryption technique that complies with the National Bureau of Standards Proposed U.S. Federal Information Processing Data Encryption Standard (DES) as defined in the Federal Register, Volume 40, Number 149, August 1, 1975, page 32395. Specify the DES parameter with the name of a PDS that contains the BQKDES module or SIP processing will not be completed.

**BOTH** Provides support for both the AET and DES encryption techniques. Specify the BOTH parameter with the name of a PDS that contains the BQKCIPH and BQKDES modules or SIP processing will not be completed.

#### *pdsname*

Specifies the name of a PDS that contains the encryption modules to be included. Specify *pdsname* with the AET, DES, or BOTH parameters or SIP processing will not be completed. If you specify a PDS name with the NO parameter, the PDS name is ignored.

**Note:** If data encryption support is included for 3614 and/or 3624 type devices, the dummy modules included with the 3600 Independent Release must be replaced by operational modules. IBM distributes object modules for BQKCIPH and BQKDES to 3614 and 3624 customers on special request.

### RES=NO|YES

This parameter specifies if the PARS/IPARS application 'RES0' is to be included in the system. In defining a subsystem other than the basic subsystem in a multiple database function (MDBF) environment, this parameter is the only way to specify the PARS/IPARS option. If RES=YES is specified, the TPF system will activate a series of RES programs. A list of these programs can be found in IBMPAL by locating all program invocations where FUNC=&SBRES is specified.

#### Notes:

1. It is the obligation of the system programmer to ensure that all of these programs are loaded into the TPF system.
2. This parameter is specific to the PARS/IPARS application and should not be confused with a general reservation type application. If the PARS/IPARS application programs are not needed this parameter should be coded as NO.

### FQTK=NO|YES

Specifies Fare Quote Program Product support. If FQTK=YES is specified, the TPF system will activate a series of Fare Quote Ticketing programs. A list of these programs can be found in IBMPAL, by locating all PAL macro invocations where FUNC=&SBNFQTK is specified.

**Note:** It is the obligation of the system programmer to ensure that all of these programs are loaded into the TPF system.

### MSGSW=NO|YES

Specifies the inclusion of the message switching function.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### USER=USA|WTC

Specifies whether the support functions for a WTC (World Trade Corp.) user are to be included or whether support is for a non-WTC (i.e. USA) user.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### ENTERPRISE=IBM|*enterprise name*

The 1-to-8 alphanumeric character string indicating the name of the enterprise or operations center associated with the system. The same name can be used to identify multiple TPF complexes within an enterprise or operations center. This name is displayed with the TPF complex name and the CPU ID in an IPL message when the system is IPLed. You can display this information in an online environment using the ZDSID command. See *TPF Operations* for additional information.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

The enterprise name is stored internally and displayed online in a right-justified field.

**The default is IBM (stored as ' IBM').**

## CONFIG

### **COMPLEX=TPF|complex name**

The 1-to-8 alphanumeric character string indicating the name of the TPF loosely coupled complex or stand-alone processor that is being generated. Unique names should be used to distinguish multiple TPF complexes within an enterprise or operations center.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

The complex name is stored and displayed in a left-justified field. **The default is TPF (stored as 'TPF ').**

### **DATEFMT=MDY|DMY|YMD**

Specifies the date format the system clocks use where **M**=month, **D**=day, and **Y**=year.

### **NEF=NO|YES**

This parameter should be coded YES if the user requires ALCI support.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### **MPIF=NO|YES**

Include the Multi-Processor Interconnect Facility. Must be coded as YES in a basic subsystem / loosely coupled generation.

Do not code this parameter as YES in a non-basic subsystem generation.

### **ACF=NO|YES**

Controls whether OSTG segments are link-edited.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### **SELECT=NO|YES**

Enables or disables the E-type loader selective activate function. If SELECT=YES, selected ECBs can enter the programs in a selectively activated loadset. Otherwise, no ECBs can enter the programs in a selectively activated loadset. For more information about the E-type loader selective activate function, see *TPF System Installation Support Reference*.

### **TPFAR=NO|YES**

Specifies whether the TPF Application Requester (TPFAR) feature is included in the system.

### **TPFDF=NO|YES**

Specifies whether the TPFDF product is included in the system initialization program (SIP).

#### **NO**

Specifies that the SIP does not include the TPFDF product in the TPF system.

#### **YES**

Specifies that the SIP includes the TPFDF product in the TPF system.

### **VEQR=NO|YES**

Specifies migration mode.

This parameter should be set to YES in a test system when migrating to the TPF 4.1 system (migration mode).

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.



**BPCRLOAD**

Controls the method for loading the core resident program area (CRPA).

**NO** The CRPA is loaded during restart while the TPF system is cycling to 1052 state.

**YES** The TPF system does not load the CRPA during restart. Rather, it loads core resident programs when they are called for the first time.

This parameter must be the same for all subsystems (BSS and non-BSS).

**Note:** Programs that are allocated as PRELOAD are always loaded during restart, regardless of the value specified for this parameter.

**DBRBUF**

Specifies if buffered tape devices are used during the database reorganization (DBR) output phase.

This parameter is ignored when generating a non-basic subsystem.

**Attention:** Running the DBR output phase with buffered tape devices will improve performance; however, if the TPF system IPLs again or tape errors occur while DBR is in progress, you may lose data.

**WTOPCUNS**

Specify one of the following:

**YES** The WTOPC message package sends remote terminals unsolicited messages.

**NO** The WTOPC message package sends remote terminals solicited messages.

This parameter must be the same for all subsystems (BSS and non-BSS).

**PREMSG=YES|NO**

Specifies whether messages generated by the system message processor will be prefixed by the CSMP0097I message.

**PROCx=(serial#, model#, SSA0, SSA1, SSA2, SSA3, SSA4, SSA5, SSA6, SSA7)**

The PROCx parameter (where x is 1–32) defines a physical processor (CPC) in a loosely coupled (LC) processor complex. In a non-LC environment, the PROCx parameters are ignored and a warning message is issued if they are coded.

In an LC processor complex, at least one PROCx parameter must be coded and, for each PROCx that is coded, at least 2 but no more than 10 subparameters are required. The first subparameter is the processor serial number and the second subparameter is the processor model number. The encoding of the processor serial number corresponds to bits 12–31 of the doubleword addressed by the STIDP instruction, while the encoding of the model number corresponds to bits 32–48 of the doubleword addressed by the STIDP instruction. Both the serial and model number subparameters are required. The serial number is specified by a 1 to 5 character hexadecimal number, while the model number is specified by the 4 character name.

The time-of-day (TOD) *Synchronization Selection Address (SSA)* is the port that other processors receive signals on when this processor is the master. The SSA is specified on the 3rd through the 10th subparameters of a PROCx statement and is only required if the TOD Clock Synchronization RPQ is installed or if an STR is using a TSC for TOD clock synchronization. The

## CONFIG

position of the third through tenth subparameters corresponds to the addresses of the CPU(s) within a processor (where *processor* refers to either a real processor or one side of a partitioned processor). See “Processor Definition” on page 31 for a discussion on partitionable processors.) Within an LC complex each TPF image must possess a unique SSA. If the TOD Clock Synchronization RPQ is installed, at least one of the SSA subparameters must be coded for each PROCx parameter that is coded. The only valid values for an SSA range from 0 to 7. Do not code the SAA subparameter for PROC9–PROC32.

For uniprocessors, only one of the eight SSAs will be coded. For multiprocessors, all CPUs in the same partition must possess the same SSA. If the SSA for a CPU is omitted, then that CPU cannot be IPLed.

For partitioned processors, the side that will be included in the LC complex must have its CPU(s) coded. If both sides of a partitioned processor are used within the same LC complex, then each partition must have a unique SSA, and each CPU within the partition must have the same SSA value.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

### **SYSID=A|v1[(v1[,...,v8])]**

One or more characters ranging from A through Z or 0 through 9 that will be used to assign the TPF processor ID(s). A system being defined as a loosely-coupled system will define more than one and up to eight TPF processor IDs (see Example 2). Specifying more than one TPF processor ID will inform SIP that this is a loosely-coupled generation. The order in which each TPF processor ID is presented to SIP through this parameter becomes the ordinal number that TPF will use to refer to each processor.

The first TPF processor ID takes on additional significance — this processor will contain the EP and BSC communication functions. Also, the system communications keypoint (SCK) generation must use the first TPF processor ID as its pilot processor ID. Finally, the name of the Application Name Table (ANT) deck produced by SIP is derived from the first parameter value. If the first processor ID listed is A, the name of the ANT deck will be ANTA. If it is B, the name of the ANT deck will be ANTB, and so on.

For a non-basic subsystem, this parameter must be coded the same as it was for the basic subsystem.

**Note:** When a loosely-coupled environment exists:

1. The MPIF feature is also required to permit IPC communications among the processors. This is accomplished by coding MPIF=YES on the CONFIG macro.
2. Multiple Systems Network Facility (MSNF) support must be included in the system.
3. 4KB blocks must be increased to account for SIPC requirements. See “CORREQ” on page 229.

## Examples

**Example 1:** The following specifies a processor with a printer address of X'00E', and the TPF processor ID is A.

```
CONFIG APRNT=00E,SYSID=A
```

**Example 2:** The following example (illustrated in Figure 29 on page 228) specifies a loosely coupled complex. Each processor in the complex has a printer address of

X'00D'. Since PROC3 defines a partitioned processor, it receives two of the four processor IDs that were specified on the SYSID parameter. The other two processors receive only one processor ID each. The actual assignment of a processor ID to a processor is done at IPL.

```
CONFIG APRNT=00D,SYSID=(A,B,C,D),
      PROC1=(10000,9021,,,1),
      PROC3=(20000,3090,,3,3,4,4),
      PROC4=(30000,9121,,2,2),
```

A 9021 has a single CPU at address 2. For the example above, this implies that an SSA should be coded on the fifth subparameter of PROC1. Subparameters three and four are left undefined. A 9121 has two CPUs, one at address 1 and one at address 2. For the example above, this implies that an SSA should be coded on the fourth and fifth subparameters of PROC4. Subparameter three is left undefined.

The physical connection of the TOD Synchronization RPQ cables must follow the SSA number as shown in Figure 29 on page 228. PROC1 has an SSA number of 01. Notice that the lines connecting PROC1 with PROC3 and PROC4 use SSR 01 whereas PROC3 Side A has an SSA number of 03 so the connections to all other CPUs use SSR 03.

**Note:** The *Synchronization Selection Register (SSR)* is the port that this processor receives signals on when it is the slave. When the high-order bit is on this processor is the slave (synchronized remotely). When the high-order bit is off, either the processor is the master for the complex or it is running in local mode. By convention the master has its SSR set equal to its SSA.

**Example 3:** The following example specifies a loosely-coupled complex that is using a Sysplex Timer (STR) as the synchronization source.

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

There are no processors with the TOD Clock Synchronization RPQ; therefore no SSA subparameters are required.

```
CONFIG APRNT=00D,SYSID=(A,B),
      PROC1=(12345,3090),
      PROC2=(67890,3090)
```

X  
X

**Example 4:** The following example specifies a loosely-coupled complex that is using an STR as the synchronization source. Processor 1 is directly connected to the STR and therefore does not require any SSA subparameters. Processors 2 and 3 both have the TOD Clock Synchronization RPQ and are connected to the STR through the TOD Synchronization Compatibility RPQ. Processors 2 and 3 require SSA subparameters due to the presence of the TOD Clock Synchronization RPQ. In addition, Processor 3 defines a partitioned processor.

```
CONFIG APRNT=00E,SYSID=(A,B,C,D),
      PROC1=(10000,3090),
      PROC2=(20000,3083,,,0),
      PROC3=(30000,3090,,1,1,2,2)
```

X  
X  
X

CONFIG

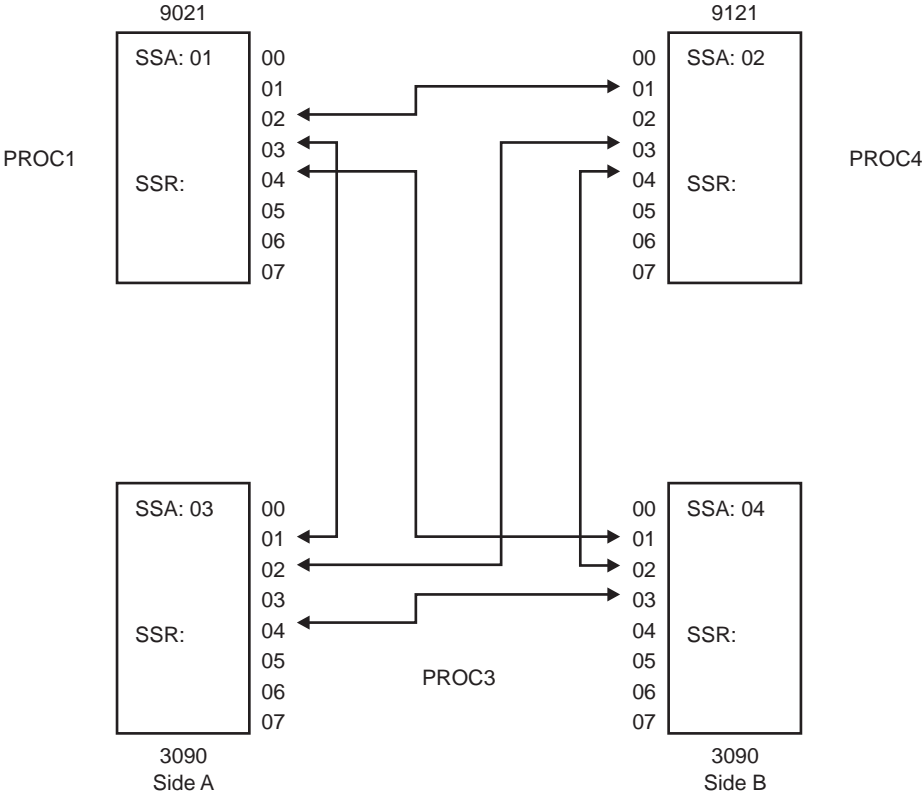


Figure 29. Sample Configuration

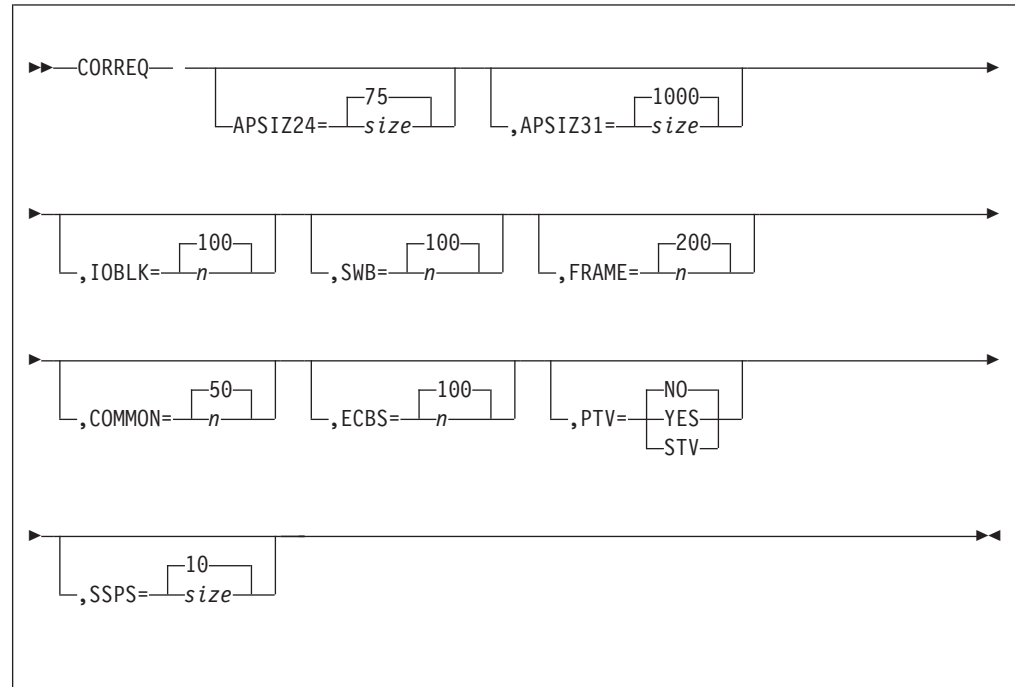
References

Mapping support in the *TPF Data Communications Services Reference*.

## CORREQ

The CORREQ macro specifies main storage requirements for the online system. Many of the values defined with this macro can be altered in the online system by issuing operator commands. See the ZCTKA command in *TPF Operations*. The CORREQ macro is required.

### Format



#### **APSIZ24=75|size**

Size (in 1KB blocks) of the program area for core resident programs that execute in 24-bit addressing mode. This area is carved below the 16MB boundary.

The sum of the values coded for APSIZ24 and APSIZ31 must be greater than or equal to 1000.

This parameter is subsystem unique. The value of this parameter is stored in keypoint A (CTKA).

#### **APSIZ31=1000|size**

Size (in 1KB blocks) of the program area for core resident programs that execute in 31-bit addressing mode. This area is carved above the 16M boundary.

In addition to core resident assembly language programs running in 31-bit mode, the application area above the line contains all C language programs. You may need to increase the size of the 31-bit core resident program area (CRPA) to accommodate E-type loader versions of C load modules. The sum of the values coded for APSIZ24 and APSIZ31 must be greater than or equal to 1000.

This parameter is subsystem unique. The value of this parameter is stored in keypoint A (CTKA).

**IOBLK=100|n**

Number of I/O Control blocks (IOBs). IOBLK must be greater than zero.

**SWB=100|n**

Number of system work blocks to be used by the online system. SWB must be greater than zero.

Some functions that require SWBs include:

**MPIF** Output messages are queued using SWBs. See “Storage Required For Control Blocks” on page 46 for more information.

**SNA** SNA uses SWBs for scheduling post interrupt routines. You need three SWBs for each CTC link. No SWBs are used for NCPs.

**TPF MQSeries**

The local queue manager uses SWBs for messages within processor unique queues.

Messages in a memory queue reside in chained SWBs. The first SWB in a message chain can contain up to approximately 300 bytes of message data and subsequent SWBs in the message chain can hold up to approximately 950 bytes of message data.

**FRAME=200|n**

Number of 4KB frames.

Keep in mind that many components in the TPF system use frames. If your TPF system uses the E-type loader, C load modules, and the system heap facility, use the following formulas to ensure that you have enough frames allocated for these components. Use data collection and reduction to assist you in determining the overall number of frames that your TPF system requires.

- Use the following formula when you are allocating frames to ensure that enough storage is available for the E-type loader to use.

1. Determine which one of the following formulas produces the highest number of frames:

– # frames =

$$\frac{((\text{maximum number of loadsets loaded at one time}) \times (\text{size of an LSD entry}))}{(\text{size of a frame})}$$

**Note:** The size of an LSD entry can be obtained from the `c$idslzd.h` header file.

– # frames =

$$\frac{((\text{maximum number of programs in a loadset}) \times (12))}{(\text{size of a frame})}$$

– # frames =

$$\frac{(((\text{number of programs allocated}) + (\text{average number of programs whose allocator characteristics have been changed})) \times (\text{size of a compacted PAT entry}))}{(\text{size of a frame})}$$

**Note:** The size of a compacted PAT entry can be obtained from the `c$idsold.h` header file. A program’s allocator characteristics

can be changed by issuing the ZAPAT command online, or by changing an allocator deck offline.

2. After determining which formula produces the highest number of frames, multiply this number of frames by the number of entries that will be doing E-type loader functions simultaneously.

For example, if one of the formulas results in 200 frames and you expect to have 2 entries performing E-type loader functions simultaneously, then you should allocate at least 400 frames that could be used by the E-type loader.

- Frames are used while bringing in a C load module from file. Once the C load module has been brought into main storage from file, the frames are returned; that is, the C load module does not remain in the frames. C load modules have the potential of being quite large. When you are allocating frames, use the following formula to ensure that there is enough storage available to bring in the C load modules from file.

$$\# \text{ frames} = ((\text{size of largest C load module}) / 4\text{KB}) \times 2)$$

Multiply the result by the number of C load modules being brought in from file at one time.

- All ISO-C storage is allocated from the pool of system frames. The amount of storage that is used by each ECB depends on the ISO-C storage parameters set in CTKA as well as the number of DLMs entered.

An ISO-C environment is created the first time an ECB enters a DLM. The number of 4KB frames that are allocated to the ISO-C stack and control blocks depends on the setting of the ISA in CTKA. The minimum value to which you can set the ISA is 2. If the stack overflow routine is called, additional frames are allocated up to a maximum that is defined by the MSHS field in CTKA.

A minimum of one 4KB frame is allocated for each DLM entered that contains static storage.

Use the following formula to calculate how much storage is allocated to the ECBs to execute the DLM:

$$\# \text{ frames} = \text{maximum \# ECBs} \times (\text{maximum stack size} / 4\text{KB} + \text{maximum \# programs containing static})$$

For example, if there are 200 ECBs and each ECB runs 10 different DLMs that all contain static, and the maximum stack size is 12KB, the number of frames needed would be:  $2600 = 200 \times (12\text{KB} / 4\text{KB} + 10)$

- In TPF systems with less than 2 GB of storage, if there is no need to remove the real storage to make room for system heap virtual addresses, frames are also used to satisfy requests for system heap storage. The maximum number of frames that can be allocated to the system heap depends on the size defined in CTKA.

Use the following formula to calculate how many frames are needed for the system heap **without** cache support:

$$\# \text{ frames} = \text{size of system heap (MB)} \times 256$$

For example, if the system heap is defined as 10 MB, 2560 is the maximum number of frames that can be used for the system heap.

To calculate how many frames are needed for the system heap **with** cache support:

1. Determine the hash table entry size:

## CORREQ

$$((pk + sk + d + 44 + 3) / 4) * 4 = e$$

where:

**pk** Primary key length  
**sk** Secondary key length  
**d** Data length  
**e** Hash table entry size

2. Determine the hash anchor area size:

$$(y * 32) = h$$

where:

**y** Number of hash entries (assume = x)  
**h** Hash anchor area size

3. Then:

$$(h + (e * x) + 8191) / 4096 = f$$

where:

**x** Number of entries  
**f** Number of system heap frames

For the file system i-node cache, the values are:

**x** user value  
**y** x (for this calculation)  
**pk** 4  
**sk** 0  
**d** 132

For the file system directory cache, the values are:

**x** user value  
**y** x (for this calculation)  
**pk** 256  
**sk** 4  
**d** 32

**COMMON=50|n**

Number of 4KB common blocks.

Should the DCL list overflow then common blocks may be depleted. An additional 5 to 10 blocks should be allocated to handle this unlikely condition.

You may need to increase the number of common blocks that are allocated for ISO-C. Common blocks are used in ISO-C to hold new versions of arrays of library addresses (AOLAs) when a loadset that contains a library is activated using the ZOLDR ACTIVATE command. Use the following formula to estimate the approximate number of additional common blocks you need.



Number of  
additional common blocks = (maximum number of active loadsets containing libraries) \* 2

The multiplication factor of 2 in the previous formula takes into account deactivating (using the ZOLDR DEACTIVATE command) loadsets that contain libraries as well as excluding libraries from active loadsets. For these, the activation number is increased, a new EAT slot is created, and a new AOLA in a common block is created.

**ECBS=100|n**

Number of 12KB ECBs.

#### PTV

Specify one of the following:

**NO** Online system not in test mode.

**YES** Online system is in test mode and program test vehicle (PTV) is present. See the *TPF Program Development Support Reference*.

**STV** Phase I/Phase III of PTV is not required. (This value is currently used by IPARS users only.)

**SSPS=10|size**

Size (in 1-MB units) of the system heap area. In TPF systems with 2 GB of storage, the system heap area is permanently backed with real storage. In TPF systems with less than 2 GB of storage, if there is no need to remove the real storage to make room for system heap virtual addresses, the system heap area is backed with 4 KB frames as each system heap storage request is made.

Common blocks must end on a megabyte boundary because frames have to start on a megabyte boundary. (Frames have to start on a megabyte boundary to limit the number of page tables required in the TPF system in order to improve performance.) If the common blocks must end on a megabyte boundary, then increasing the amount of storage to, for instance, 14MB or 16MB may push the common block area above a megabyte boundary. As a result, more common blocks would be created to fill the megabyte area remaining. When the common block area increases, VFA decreases. The other block types increase in proportion as storage increases; the common blocks do not because of the megabyte alignment.

For more information about the CORREQ macro, see “TPF Main Storage Layout” on page 33.

## Examples

None.

## References

None.

---

**CRASTB**

The CRASTB macro is used to specify the console device that is to be initially used as the control program operator terminal (also referred to as Prime CRAS set, or PRC). This device may be a locally attached 3270 (including the 3270 native console), a 1052, or a 3215. If a 3270 is selected, a 3270-type printer must be specified in the ROCRS operand unless the no RO parameter specifies no RO is desired in the system. If a 1052 or 3215 is selected, an alternate console device to be used as a fallback device for the operator's terminal may also be specified.

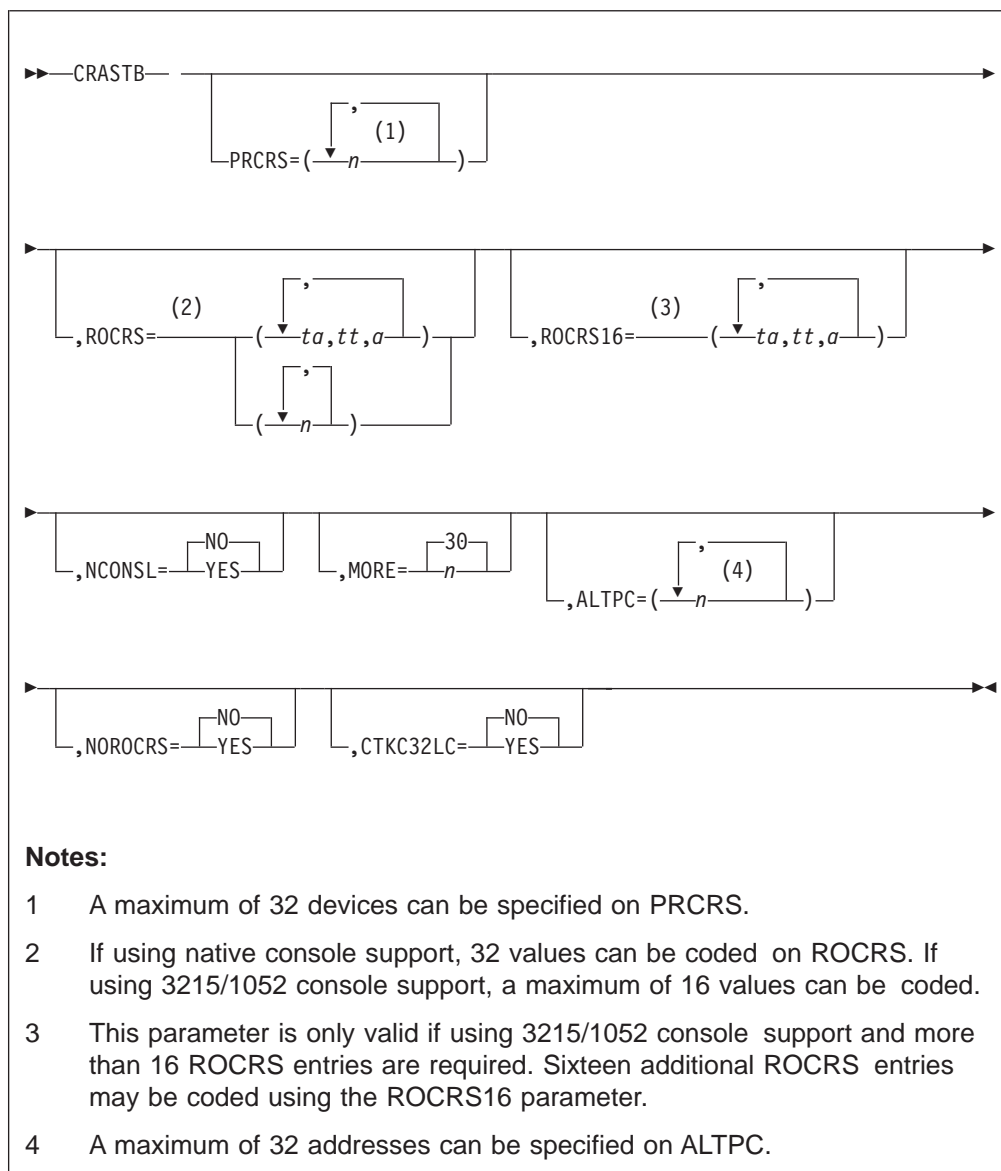
In addition a terminal address may be specified to designate the terminal to which the control program outputs noncritical error messages (referred to as Receive only CRAS set—ROCRS).

After the system has been cycled to 1052 state, the PRC and ROCRS addresses can be altered by issuing commands. However, the IPL program and the in-core dump formatter program (ICDF) will continue to use the console device address specified by this macro for communications with the operator.

This macro provides input to keypoint record C (CTKC). Refer to data macro CK8KE.

The CRASTB macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



### PRCRS=*n*

Two-digit subchannel address (in hex) of the console device to be assigned as the primary CRAS operator terminal. The subchannel address must be between x'01' and x'FF'. The device must be a 1052, a 3215, or a locally attached 3270-type terminal (this includes the 3270-type native console). Regardless of actual device characteristics, the 3270 PRC will be supported as a 3270 Model 2. A maximum of eight devices are allowed. The following device types are supported as a 3270 type PRC:

3278 models 2, 3, 4.

#### Notes:

1. When PRCRS is a locally-attached 3270, ROCRS format 2 must be specified (if the NOROCRS parameter is coded as YES, this parameter is not required), and the NCONSL parameter must be coded as YES. All subchannel addresses coded will be available to all processors serving as

## CRASTB

fallback devices. All processors start at the top of the PRCRS list and search for the first available terminal. This is assigned as the primary CRAS operator terminal of the system.

2. In native console systems fallback consoles are coded with the PRCRS parameter instead of the ALTPC parameter. There may be more consoles coded than there are processors being generated, according to the SYSID parameter of the CONFIG macro. This is not true in non-native console systems. In non-native console systems the number of system consoles must exactly equal the number of processors being generated because fallback consoles are coded with the ALTPC parameter.

**ROCRS**=(*ta,tt,a,ta,tt,a,...ta,tt,a*)

*ta* The line, interchange and terminal address of the receive-only CRAS device in the format LNIATA where:

**LN** The two-digit symbolic line address in hex.

**IA** The two-digit interchange address in hex.

**TA** The two-digit terminal address in hex.

All CRAFTB line numbers must be within LC (local 3270) ALC or 3270 SDLC line range as obtained from LINES macro (HSALC+1).

Must be the first parameter of the triplet (*ta,tt,a*). See above.

*tt* The device type of receive-only CRAS set. Must be one of the following keywords:

**1977** A 1977

**97UN** A 1977 with unsolicited message light

**97WT** A 1977 with TAB feature

**97UT** A 1977 with unsolicited message light and TAB

**8021** A 1980-21

**8024** A 1980-24

**1052** A 1052 or 3215

**84L1** A 3284/3286 model 1 on an LC line.

**84L2** A 3284/3286 model 2 on an LC line.

Must be the second parameter of the triplet (*ta,tt,a*). See above.

*a* A character (A-Z or 0-9) to identify the system in a message routing network to which the CRAS set is attached. If omitted, the default is the corresponding TPF system ID coded in CONFIG(SYSID).

Must be the third parameter of the triplet (*ta,tt,a*). See above.

A maximum of 32 devices is permitted.

**Note:** If you want to use the same console device as specified for the PRC for the ROCR device, code the device as 1052.

If the device is 1052, the LNIATA is set to a default value of 010002 and can be omitted.

If the ROCRS parameter is omitted there will be a default entry, consisting of an LNIATA of 010002 and a device type of 1052, created for every PRCRS entry. If any pair (ta,tt) of entries is defaulted, the default for that entry will be an LNIATA of 010002 and a device type of 1052.

**ROCRS=(*n,n,...n*)**

This format must be used when the PRC is a locally-attached 3270, unless the NOROCRS parameter is coded as YES.

*n* Two-digit subchannel address (in hex) of the locally attached ROCRS 3270 printer. The subchannel address must be between x'01' and x'FF'. Regardless of the actual device characteristics, the device will be supported as a 84L2 (reference format 1). Although each processor's RO is positionally determined, all subchannel addresses coded will be available to all processors serving as fallback devices.

A maximum of 32 devices is permitted.

**Note:** In a system with 3270 native console support, the number of PRCRS entries must be equal to the number of ROCRS entries defined. If the NOROCRS parameter specifies YES, then the ROCRS parameter does not need to be coded.

**ROCRS16=(*ta,tt,a,ta,tt,a,...ta,tt,a*)**

Only use this parameter if more than 16 ROCRS entries are required when using 3215/1052 console support. Code the *ta,tt,a* value sets using the same rules as for the ROCRS parameter.

**NCONSL=NO|YES**

Specifies 3270 native console support. If this parameter is YES, then ROCRS format 2 must be specified, and the ALTPC parameter must be defaulted or zero. If the NOROCRS parameter specifies YES, then the ROCRS parameter does not need to be coded.

**Note:** Inclusion of native console causes 3272 local support to be included in the system.

**MORE=30|*n***

Specifies, in decimal, the number of seconds before MORE.... is cleared from the 3270 local screen, allowing queued messages to be printed. This parameter is valid for both native and non-native console support.

**ALTPC=00|(*n,n,...n*)**

Two-digit subchannel address (in hex) of an alternate 1052 or 3215 console to be used if the 1052 or 3215 PRC device is not functioning. When a 3270 console is selected as PRC, no alternate is provided and this value must be (and can be defaulted to) zero (00). Association with a prime CRAS is positionally determined.

A maximum of 32 alternate consoles is permitted.

**NOROCRS**

Specifies whether or not a 3270 RO printer will be attached to each processor.

**YES** The TPF system will bypass checks for an RO printer, and will not allow operators to specify the RO parameter on any ZACRS commands. Prime CRAS output will not be echoed to the RO printer, even if the RO device is physically attached to the TPF system. If no RO device is

## CRASTB

attached to the system, then the IPLB0032I NO 328X PRINTER IS OPERATIONAL - RESTART CONTINUES message will not appear on the Prime CRAS.

**NO** TPF processing of the 3270 RO will continue as usual. This parameter is only relevant for a 3270 (native) console environment.

### CTKC32LC

This parameter specifies whether CTKC should be in 32-way loosely coupled format.

**YES** CTKC is generated with the 32-way loosely coupled processor support flag (CK8LC32) set on, indicating that CTKC is in 32-way loosely coupled format. YES must be coded if one of the following is true:

- This is an existing loosely coupled complex and the ZMIGR command with the CTKC and CONVERT parameters specified has been completed successfully.
- This is a new complex with 32-way loosely coupled processor support on all processors.

**NO** CTKC is generated with the 32-way loosely coupled processor support flag (CK8LC32) set off, indicating that CTKC is not in 32-way loosely coupled format. NO must be coded if one of the following is true:

- This is an existing loosely coupled complex and the ZMIGR command with the CTKC and CONVERT parameters specified has not been completed successfully.
- The ZMIGR command with the CTKC and FALLBACK parameters specified has been completed and CTKC is not in 32-way loosely coupled format.

## Examples

**Example 1:** The system being generated has two 3215 type console devices with addresses of X'09' and X'1F'. Device X'09' is to be the PRC device and X'1F' is to be the fallback console device. The PRC device is to be also used as the ROCRS device.

```
CRASTB      PRCRS=09,ALTPC=1F,NCONSL=NO
```

**Example 2:** The system being generated has three 3215 type console devices with addresses of X'09', X'0A', and X'1F'. Devices X'09' and X'0A' are to be the PRC devices and X'1F' will be the fallback console device for X'0A'. X'09' has no fallback console device. The ROCRS device for X'09' will be LNIATA of 010003 and device type of 8021. For PRC X'0A' the PRC will also be used as the ROCRS device.

```
CRASTB      PRCRS=(09,0A),ALTPC=(,1F),ROCRS=(010003,8021,B)
```

**Example 3:** The same as the example immediately above except: for PRC X'09' the PRC will also be used as the ROCRS device. The ROCRS device for X'0A' will be an LNIATA of 010003 and device type of 8021. All subchannel addresses coded will be available to all processors serving as fallback devices. A maximum of eight is permitted.

```
CRASTB      PRCRS=(09,0A),ALTPC=(,1F),          *
            ROCRS=(,A,010003,8021,B)
```

**Example 4:** The system being generated has two locally attached 3270-type console devices with subchannel addresses of X'91' and X'94' to be used as the PRC devices. The corresponding locally-attached ROCRS devices have subchannel addresses X'93' and X'92'. Subchannel address X'95' defines an additional PRC

fallback device and subchannel address X'96' defines an additional RO fallback device. MORE.... is cleared from the screen every 35 seconds.

```
CRASTB  PRCRS=(91,94,95),ROCRS=(93,92,96),      *
        NCONSL=YES,MORE=35
```

If two processors exist in this system, the first is defined as having its PRC on subchannel 91 with subchannels 94 and 95 as alternates and its RO on subchannel 93 with subchannels 92 and 96 as alternates. The second processor is defined as having its PRC on subchannel 94 with subchannels 91 and 95 as alternates and its RO on subchannel 92 with subchannels 93 and 96 as alternates.

**Example 5:** The same example 4 above EXCEPT both processors have PRCs on subchannel 91 and alternate PRCs on subchannels 94 and 99. Additionally, both processors have their ROs on subchannel 96 and alternate ROs on subchannels 92 and 95.

```
CRASTB  PRCRS=(91,91,94,99),ROCRS=(96,96,92,95),  *
        NCONSL=YES,MORE=35
```

**Note:** The number of ROCRS devices specified must equal the number of PRCRS devices specified. Therefore, to specify additional PRC (or RO) fallback devices with no additional RO (or PRC) devices, duplicate ROCRS (or PRCRS) devices must be coded (see following example).

**Example 6:** The same example 5 above EXCEPT subchannels 84 and 85 are additional PRC fallback devices. (Note that subchannel 95 is duplicated on the ROCRS parameter to correspond to the additional PRCRS devices.)

```
CRASTB  PRCRS=(91,91,94,99,84,85),                *
        ROCRS=(96,96,92,95,95,95),NCONSL=YES,MORE=35
```

## References

None.

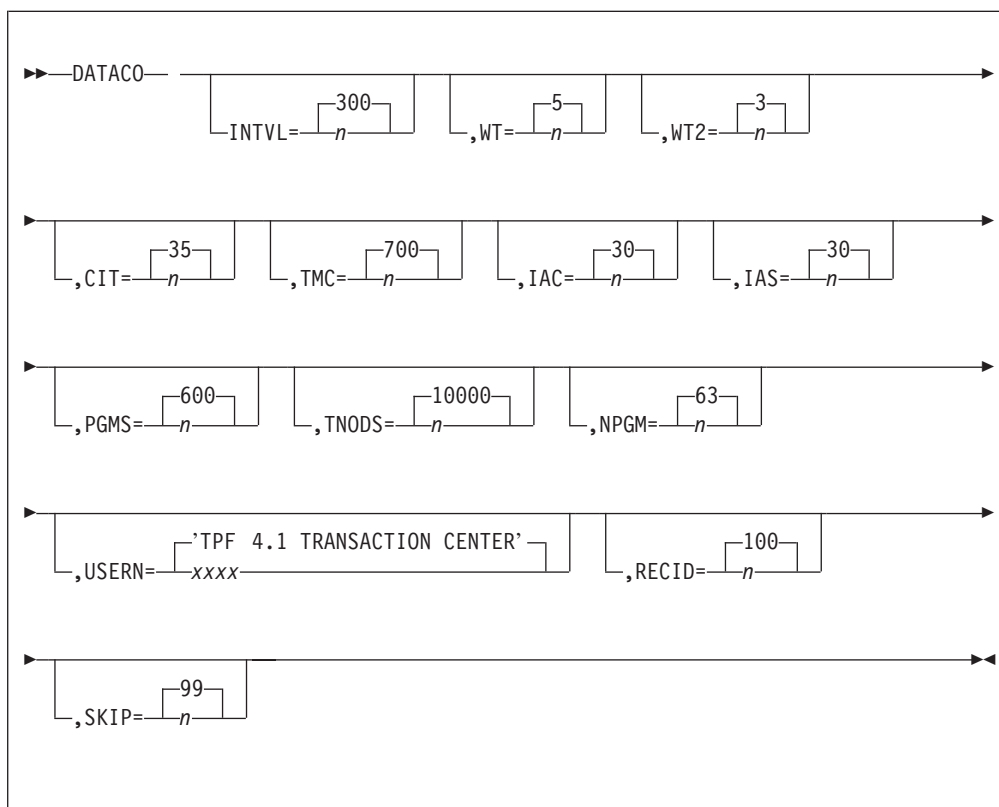
# DATA CO

The DATA CO macro is used to specify values used by the TPF data collection and reduction package. Some of these values will become part of the data reduction pre-compiler segment (JPC0) that is used in compiling the other data reduction segments. See *TPF System Performance and Measurement Reference* for information about the pre-compiler segment JPC0.

This macro need not be coded if all of the default values shown here are acceptable. SIP automatically assigns these values if the macro is omitted, although a warning message is issued.

The DATA CO macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



### INTVL=300|n

The number of sampling intervals for the life of Data Collection in the continuous mode. Also, the maximum number of sampling periods for the life of Data Collection in the sampling mode. INTVL may not exceed 300; it must be greater than 0.

### WT=5|n

Weighting factor to be applied to low-speed messages. This is the number of high-speed messages that is equivalent to one low-speed message.



**WT2=3|n**

Weighting factor to be applied to routed messages. Valid ranges of WT2 values are 1 to 9. Note these values will generate a definition containing a decimal point (i.e., 3 will be generated as .3).

**CIT=35|n**

Specifies the maximum number of cities in the system that are associated with an interchange address.

**TMC=700|n**

Specifies the maximum number of terminals per city.

**IAC=30|n**

Specifies the maximum number of interchange addresses per city. The maximum allowable is 32. The minimum is 1.

**IAS=30|n**

Maximum number of unique LN (lines) and IA's (interchange addresses) in the system. The minimum value is 1.

**PGMS=600|n**

Number of programs to be reported. The minimum value is 1.

**TNODS=10000|n**

Specifies the total number of resources (both local and nonlocal) in the network. This number must be equal to or greater than the number declared for the MAXRVT keyword of the SNAKEY macro coded at network definition time.

**NPGM=63|n**

n = number of program names that can be placed in the SM record on the RTC tape. The value specified must be between 1 and 63.

**USERN='TPF 4.1 TRANSACTION CENTER'|xxxx**

A 1–25 character name describing the user for which the report is being produced. If blanks are part of the character string, it should be enclosed in apostrophes.

**RECID=100|n**

Maximum number of unique pool and fixed record IDs (byte 1 and 2 of record) for data to trace. The minimum value is 2 and the maximum is 500.

**SKIP=99|n**

Sets the default value for the skip parameter on the ZMEAS command. The minimum value is 0 and the maximum is 9999. See the description of the ZMEAS skip parameter in *TPF Operations* for more information.

## Examples

The following shows the DATA CO macro for a system whose report heading is to be TPF TEST CENTER with 200 sampling intervals, 50 terminals per city, and a default value of 0 for the skip parameter on the ZMEAS command. All other options default.

```
DATA CO INTVL=200,USERN='TPF TEST CENTER',TMC=50,SKIP=0
```

## References

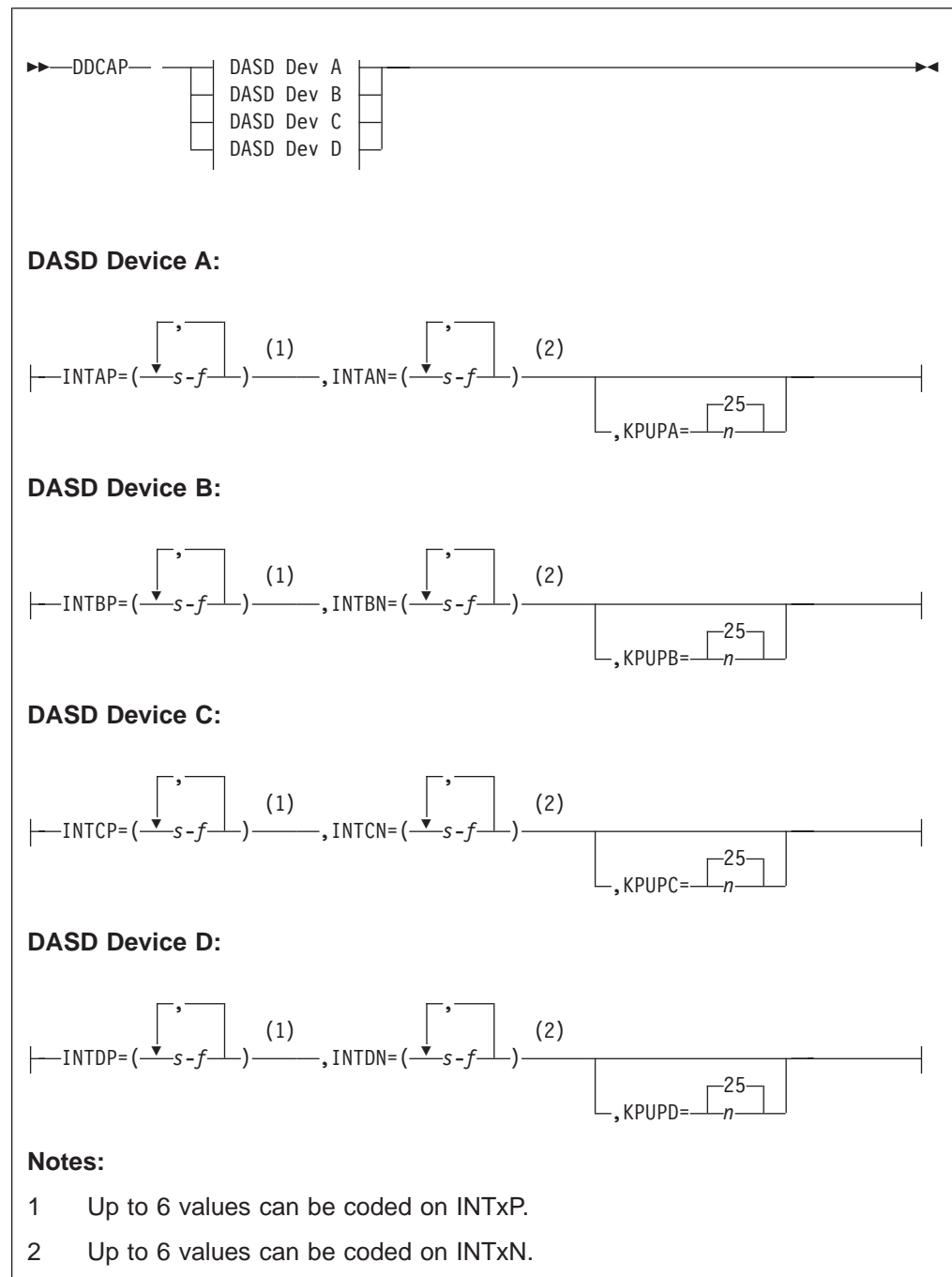
None.

### DDCCAP

The DDCCAP macro is used to establish the disk device control table for capture/restore. Parameters may be submitted for up to four DASD devices.

DDCCAP is required if the capture/restore facility is desired. If coded, parameters for at least one device must be entered. For more information about capture/restore and a detailed explanation of how to initialize these fields, see the *TPF Database Reference*.

## Format



**Note:** Fields in this macro are identical in definition for the various device types. One explanation of the parameters is provided here, where x in the keyword parameters is interpreted as follows:

Device A	-	-	-	x = A
Device B	-	-	-	x = B
Device C	-	-	-	x = C
Device D	-	-	-	x = D

## DDCCAP

### INTxP=(s-f)

From 1–6 valid disk module ranges to be captured or restored at one time where both nonduplicated and primary copies of duplicated records will be selected.

*s* A 1-4 character decimal number that defines the starting relative module number of the range, and is less than 3998.

“.” Delimiter.

*f* A 1-4 character decimal number that defines the ending relative module number of the range, and is less than 3998. For example: 0-4, 5-8, 9-15, 1600-2000.

**Note:** *s* value must be less than or equal to *f* value.

0 is the first module number. The number of mods - 1 is the highest module number (e.g., 0–9, 10–16) is invalid for a 16 module device type).

Ranges need not be submitted in any sequence. See the description of fields 'BXILOM' and 'BXIHIM' in assembled program segment BXAX.

### INTxN=(s-f)

From 1–6 valid disk module ranges to be captured or restored where only nonduplicated records will be selected. Rules for coding this parameter are identical to those stated above. See the description of fields 'BXILOM' and 'BXIHIM' in assembled program segment BXAX.

### KPUPx=25|*n*

Capture/restore keypoint file increment. This number is the interval (expressed as number of cylinders) after which the capture/restore keypoint will be filed. The allowable range, in decimal, is 1–255, however, 10 is considered too frequent. See the description of field 'BXACINC' in assembled program segment BXAX.

**Note:** Module ranges submitted for the above parameters must be consistent with ranges calculated from the SIP ONLFIL macro and must not overlap within a device type. Module ranges submitted in the primary/nonduplicated parameter must not overlap ranges in the nonduplicated only parameter.

Either primary (INTxP) or nonduplicated (INTxN) ranges must be specified for a particular device type. However, if only primary ranges are desired, the nonduplicated parameter may be omitted and vice versa.

It is important to note that the working storage required for capture is significant and in a small system the danger of a polling shutdown exists.

## Examples

The example system is defined to have 2 DASD device type A and 2 DASD device type B. All modules are to be captured and restored. The nonduplicated and primary copies of the duplicated records are to be selected. Tape speeds are 22 milliseconds. Code the DDCCAP as follows:

```
DDCCAP      INTAP=(0-1),                      X
             INTBP=(0-1)
```

## References

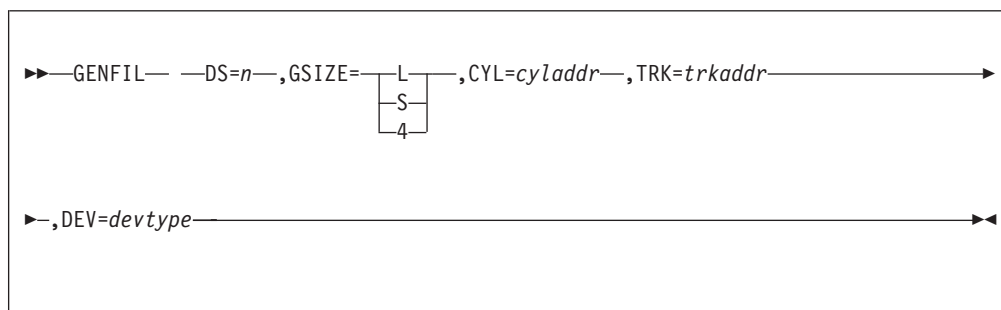
None.

## GENFIL

One GENFIL macro specifies the type and format of one general file data set. Refer to “General Data Sets” on page 77 and “General Files” on page 78 before providing input to this macro. Information from this macro is used to create the general file module table (GFMT).

The GENFIL macro is required.

## Format



### DS=*n*

Relative location in the general file module table of this entry. It corresponds to the data set number used in the general file mount and dismount messages. The valid range is 0 to 59. For more information, see *TPF Operations*.

The general file data sets which are used by the system and should be defined by GENFIL, if applicable, are:

- 2 For DASD recoup general file - large records
- 3 For DASD pool general file - large records

### GSIZE

Specify one of the following:

- L** Indicates large records in data set.
- S** Indicates small records in data set.
- 4** Indicates 4KB records in data set.

### CYL=*cyladdr*

Starting cylinder address of the data set on the general file.

### TRK=*trkaddr*

Starting head address of the data set on the general file.

### DEV=*devtype*

Specifies one of the supported DASD device types as follows:

- 3350** Model 2 or Model 2F
- 3375** Model 1 or Model 1F
- 3380** All models
- 3390** All models
- 9345**

## GENFIL

### Notes:

1. SIP produces format decks for the online modules and the loader general file. To initialize general file modules, use the MVS utility ICKDSF and the TPF formatter.
2. 3390 devices are set at installation time to either native 3390 mode or 3380 emulation mode. A 3390 running in 3380 emulation mode must be coded as DEVICEx=3380 for both types of emulation. TPF supports 2 types of 3380 emulation.
  - a. Track compatibility emulation, which is supported on 3990 control units that are not using CFLF. For sample JCL to set or change the mode of a 3390, see the description of the ICKDSF INSTALL command (SETMODE option) in the *DSF User's Guide and Reference*.
  - b. Record format emulation, which is used with CFLF. 3380 emulation in this case is set by using Licensed Internal Code (LIC) that runs directly on the 3990.

## Examples

For a system having DASD pools and using 3380 disks as general files for all functions, the GENFIL macros are coded as follows:

```
GENFIL DS=0,GSIZE=S,CYL=1,TRK=0,DEV=3380
GENFIL DS=2,GSIZE=L,CYL=60,TRK=1,DEV=3380
GENFIL DS=3,GSIZE=L,CYL=0,TRK=1,DEV=3380
```

## References

None.

---

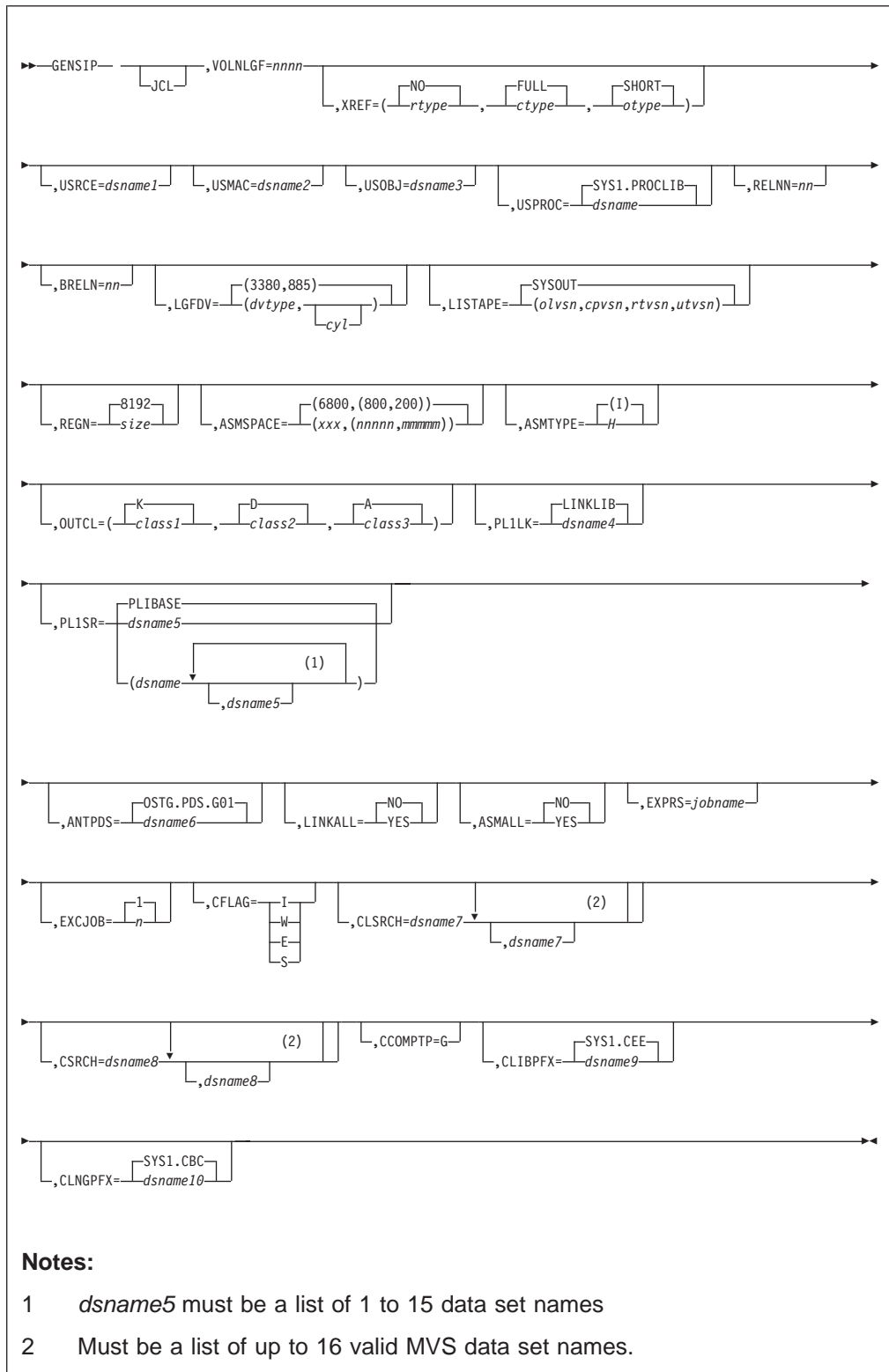
## GENSIP

The GENSIP macro performs the following functions:

- Allows the user to select the type of SIP run to be performed which establishes the interaction between the various data sets, volumes and devices that are involved in processing SIP Stage II.
- Ties together all SIP parameters that were specified during preparation of Stage I. In addition, GENSIP diagnoses the incompatibilities between the various parameters and produces the job stream that will be executed in Stage II.
- Allows the user to specify a partial SIP generation.

The GENSIP macro is required; it must be the last SIP Stage I input macro.

# GENSIP Format



## JCL

Specifies that Stage II JCL only be generated. Stage II will be nonexecutable when this parameter is used. Note that JCL is a positional parameter and must be coded as the first operand when used. **The default is a complete Stage II.**



**CAUTION:**

**A production SIP Stage I run must not use this parameter.**

**VOLNLGF=nnnn**

Specifies the numeric portion of the loader general file volume serial number. It may be a 1–4-digit decimal number, but any number shorter than four digits will be padded with leading zeros. The value of VOLNLGF is concatenated to the character portion which is passed via the VSNCHAR parameter of the ONLFIL macro. The resulting string is formatted as the LGF VSN in Stage II processing.

**Note:** The LGF VSN is always generated in the AANNNN format, regardless of the migration driver status. (See the OLDVSN parameter of “ONLFIL” on page 282 for information about activating the migration driver.)

**XREF=(NO|rtype,FULL|ctype,SHORT|otype)**

This parameter allows the user to specify the type of cross reference listing desired for **each** of the program types assembled or compiled during execution of SIP Stage II. Each cross reference may be specified as either SHORT, FULL, or NO.

*rtype* Specifies the cross reference option desired for real-time (also referred to as E-type) programs. Also included are keypoints as well as real-time special, user real-time, C language real-time, and C language user real-time programs.

For C language programs, there is no differentiation made for FULL vs. SHORT cross references. Either FULL or SHORT will instruct the C compiler to produce a cross reference.

*ctype* Specifies the cross reference option desired for the control program CSECTs.

**Note:** If the multiple database function (MDBF) is in the system, this parameter is only valid for the basic subsystem (BSS) generation. (BSSGEN=YES in the SSDEF macro.)

*otype* Specifies the cross reference option desired for offline program assemblies.

**USRCE=dsname1**

Specifies an optional user real-time (UR) and/or C language user real-time (CUR) source library to be included in the Stage II assemblies. If this operand is included, *dsname1* must be the name of a cataloged data set. **The default is no user source library referenced.**

**Note:** BLKSIZE cannot exceed 400.

**USMAC=dsname2**

Specifies an optional user macro library to be included in the SYSLIB data definition for the Stage II assemblies. If this operand is included, *dsname2* must be the name of a cataloged data set. **The default is no user macro library will be referenced.**

**Notes:**

1. BLKSIZE can not exceed 400.
2. This parameter is not used for C language compiles.

**USOBJ=dsname3**

Specifies an optional user object library to be included in the link-edit steps of

Stage II. If this operand is included, dsname3 must be a cataloged data set.  
***The default is no user object library will be called.***

**Note:** BLKSIZE can not exceed 400.

**USPROC=SYS1.PROCLIB|dsname**

Specifies the data set to contain the procedures generated for RT, UR, CRT, CUR, OL and CP MASM assemblies and compilations. The procedure to compile offline PL/I segments also resides here. Any valid qualified MVS data set name can be used. This data set must be allocated and cataloged by the user before running SIP.

**RELNN=nn**

Indicates release level if different than release level of GENSIP macro being used. ***The default is the two-digit release number.*** For example, nn is 24 for TPF 2.4.

**BRELN=nn**

The release level used for the basic subsystem generation. This parameter is to be provided only for the generation of a nonbasic subsystem generation in a MDBF environment and will be ignored if coded for the generation of a basic subsystem. ***The default is the same as RELNN.***

**LGFDV=(3380,885)|(dvtype,[cyl])**

Specifies the device type of the loader general file volume. For more information about valid device types, see *TPF Migration Guide: Program Update Tapes*. If the device type is coded but the number of cylinders is not then the number of cylinders defaults to the total number of cylinders on the device.

**LISTAPE=SYSOUT|(olvsn,cpvsn,rtvs,utvs)**

Specifies the volume serial number of the magnetic tape to be used to contain the listings produced by the Stage II assemblies using the multiple assembler program (MASM). If any of the following variables are not coded, a scratch tape will be used. ***The default is the SYSOUT printer if no LISTAPE operands are coded.***

- olvsn* This volume serial number will be used for the offline program listings.
- cpvsn* This volume serial number will be used for the control program listings.
- rtvs* This volume serial number will be used for the assembler and C language real-time program listings.
- utvs* This volume serial number will be used for the assembler and C language user real-time program listings.

The volume serial numbers are all positional and may be coded as a specific value (i.e 123456 or ABCDEF), as SCRATCH or left uncoded. A combination of the three may also be coded. However, SYSOUT may not be coded together with a serial number.

**Examples:**

LISTAPE=(,,)	4 scratch tapes will be used
LISTAPE=(,ABCDEF,SCRATCH)	3 scratch tapes for OF (olvsn),
	RLT & CRT (rtvs) and
	UR & CUR (utvs) and
	volume ABCDEF used for CP (cpvsn)

**Note:** A tape will always be assigned for each program type. The actual request for the tape will be made only if the procedure is invoked in Stage II.

**REGN=8192|size**

Specifies the region size (in 1KB units) to be used by the assembler and PL/1 compiler for object code generation. **The default is 8192**, which will be the minimum region size for Stage II assemblies. If the user specifies REGN as less than 8192, the input will be ignored.

**Example:**

```
GENSIP LISTAPE=LIST01,REGN=8192
```

**ASMSPACE=(6800,(800,200))|(xxx,(nnnnn,mmmmm))**

Specifies the size of the MVS SYSUT1 dataset for TPF system and object code assemblies. ASMSPACE corresponds to the SPACE parameter on the DD card for the SYSUT1 dataset, but it supports only the TRK, CYL, blocklength, primary quantity, and secondary quantity parameters. For additional information, please consult *MVS/ESA JCL Reference*.

Whenever a coding error is encountered with ASMSPACE, the default is taken.

**xxx** This parameter refers unit of space reserved for the SYSUT1 dataset. It may be coded as TRK, for tracks, or CYL, for cylinders, or as a decimal number, for a block size. If block size is chosen, then the number must fall in the range 1–65535 inclusive. If ASMSPACE is coded, then this subparameter must be coded.

**nnnnn** This is a nonzero decimal number representing the primary quantity of the units described by the first subparameter. If ASMSPACE is coded, then this subparameter must be coded.

**mmmmm**

This is a decimal number representing the secondary quantity of the units described by the first subparameter.

**ASMTYPE=(I)|H**

Specifies which assembler to use: the H assembler or the high-level assembler (HLASM). **The default is I, which is the HLASM.**

**OUTCL=(K|class1,D|class2,A|class3)**

Indicates the print output classes for the MVS SYSOUT parameters to be used when assembling CP, RT, UR and OL programs, or when compiling CRT or CUR programs. A class is an alphameric character (A–Z, 0–9) that corresponds to a type of unit record device, specified by an installation.

**class1** Indicates the print output class for control program (CP) CSECTs. If the MDBF feature is present in the system then this parameter is only valid for the basic subsystem generation (BSSGEN=YES in SSDEF macro.)

**class2** Indicates the print output class for all types of real-time programs. Generated keypoints also use this class.

**class3** Indicates the print output class for offline (OL) programs.

**Note:** Any class may be omitted provided positional integrity is kept. Following are some examples showing valid forms:

OUTCL=(B,,C)	CP=B,RT=D,OL=C
OUTCL=(B,A)	CP=B,RT=A,OL=A
OUTCL=(,K)	CP=K,RT=D,OL=K
OUTCL=B	CP=B,RT=D,OL=A

**PL1LK=LINKLIB|dsname4**

Specifies the PL/1 compiler library. If this operand is included, *dsname4* is automatically prefixed with 'SYS1.'. The resulting data set name must be a cataloged data set.

**Note:** BLKSIZE cannot exceed 400. Do not code SYS1. It will be generated by the system.

**PL1SR=PLIBASE***|dsname5<sub>1</sub>| (dsname5<sub>1</sub> [,dsname5<sub>2</sub>,...,dsname5<sub>15</sub>])*

Specifies the PL/I subroutine library or libraries. When *dsname5* is coded, it must be a list of 1 to 15 data set names separated by commas. Each name is prefixed automatically with 'SYS1.'. The resulting data set name must be a cataloged data set.

The data sets will be concatenated in the same order as they appear in *dsname5*.

**Note:** BLKSIZE cannot exceed 400. Do not code SYS1. It will be generated by the system.

**ANTPDS=OSTG.PDS.G01***|dsname6*

Specifies the data set to be used to contain the offline SNA table generation program application name table definitions. See *TPF ACF/SNA Network Generation*. Any valid qualified MVS data set name can be used. This data set must be cataloged. The string *.BSS* will be automatically appended to the end of the data set name coded in the basic subsystem of a multiple database function (MDBF) environment.

**Note:** This parameter is only to be coded when generating either:

1. A base only system or
2. The basic subsystem in a multiple database function (MDBF) environment.

**LINKALL=NO|YES**

Specifies whether SIP will send link-edited jobs to a stage II deck for a C load module.

**NO** Specifies that SIP will send link-edited jobs to a stage II deck for all programs where NO is specified for the LINK parameter on the SPPBLD macro. ***This is the default.***

**YES** Specifies that SIP will send link-edited jobs to a stage II deck for all programs where ICL is specified for the TYPE parameter on the SPPBLD macro, regardless of how the LINK parameter on the SPPBLD macro is coded.

See "Modifying the Program Tables" on page 181 for more information about the SPPBLD macro.

**ASMALL**

This parameter allows the user to override the OBJ parameter of the SPPGML (program list) macro. (See "Modifying the Program Tables" on page 181).

**NO** Specifies that the assembly process is to be controlled by the OBJ parameter of the SPPGML macro.

**Note:** ASMALL=YES is coded only when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

**YES** Specifies that all programs defined in the SPPGML macro are to be assembled or compiled and that the OBJ parameter is to be ignored.

**Note:** There is an exception. Programs with the TYPE=OCO designation are not affected by ASMALL=YES.

**EXPRS=jobname**

This parameter allows users to define the job in Stage II that will be generated by Stage I. By specifying this parameter, only the jobs named will be generated. The valid job names are F2, G1, G2, G4–G6, H, I1–I3, S, JA, JB, JC, JD, JE, JF, J1–J10, K, L1–L6, M, and S. Specifying job name M will cause the SIP report to be generated. All other job names are defined in Appendix C, “SIP Stage II Job Summary” on page 473. **The default is all jobs.**

**Example:**

If a user only wanted to generate the first three jobs in the SIP, (SIPF1A, SIPF1B, and SIPF2A), the EXPRS parameter would be coded as follows:

```
EXPRS=F
    or
EXPRS = (F2,G1)
```

Multiple job steps can be defined by separating each step name with a comma and enclosing the parameters in parentheses.

**Note:** Extreme care must be taken when using this parameter because of the dependencies of each of the Stage II jobs.

**EXCJOB=1|n**

This parameter specifies the number of MASM execute statements per job in the real-time section J4 and the user real-time section J7. This parameter is only useful when listings are printed to tape. It reduces the number of jobs produced thereby decreasing the number of tape mounts.

**Note:** When printing a large number of programs, use a tape with the largest format possible for your tape device to insure that the programs will fit on one tape. Also, it is recommended that you mount each tape as private to minimize the number of tape mounts.

**Example:**

```
mount device,vol=(SL,label),use=private
```

**CFLAG**

This parameter specifies the C language compile-time FLAG option, which determines the minimum severity of error that requires a message to be generated.

- I** List all messages
- W** List all except information messages
- E** List all except warning and information messages
- S** List only severe error and unrecoverable error messages

If the parameter is omitted, then the NOFLAG default is set.

**CLSRCH=dsname7<sub>1</sub>[,dsname7<sub>2</sub>,...,dsname7<sub>16</sub>]**

This parameter specifies the C language compile-time LSEARCH option. The LSEARCH option directs the preprocessor to look for the user #include files in the libraries. The arguments to CLSRCH must be a list of up to 16 valid MVS data set names. These data sets will be added the USERLIB DD card of the C compilation procedure created by Stage I. They will be concatenated in the order specified on the CLSRCH parameter (the first subparameter will be the first data set in the USERLIB concatenation list). These data sets must be cataloged.

## GENSIP

**CSRCH**=*dsname8<sub>1</sub>*[,*dsname8<sub>2</sub>*,...,*dsname8<sub>16</sub>*]

This parameter specifies the C language compile-time SEARCH option. The SEARCH option directs the preprocessor to look for the system #include files in the libraries. The argument to CSRCH must be a list of up to 16 valid MVS data set names. These data sets will be added to the SYSLIB DD card of the C compilation procedure created by Stage I. They will be concatenated in the order specified on the CSRCH parameter (the first subparameter will be the first data set in the SYSLIB concatenation list). This data set must be cataloged. See the *TPF Application Programming* for more information.

**Attention:** Do not add either the VM or MVS C language header data set to SYSLIB. These data sets have been deliberately omitted from the SIP generated C compilation procedures. The VM and MVS headers provide some functions which are incompatible with a TPF environment. If these headers were added, then a C program segment might compile without errors or warnings, but it would be unusable in a TPF environment.

### CCOMPTP

This parameter specifies the compiler that will be called to compile the TPF C language programs.

**G** Causes an IBM OS/390 C/C++ compiler to be called from the partitioned data set (PDS) specified with the CLNGPFX parameter.

**Note:** The SIP Stage II deck will **not** include JCL to compile TARGET(TPF) code (whether it exists or not) because OS/390 compilers do not support TARGET(TPF) code. If you have TARGET(TPF) code, you are responsible for compiling it using an MVS compiler **outside** the SIP process.

**CLIBPFX**=SYS1.CEE[*dsname9*]

This parameter specifies the prefix name of each PDS that contains the run-time library that is used by the C compiler. **The default is SYS1.CEE, which is the LE runtime library.**

**CLNGPFX**=SYS1.CBC[*dsname10*]

This parameter specifies the prefix name of each PDS that contains the C compiler. **The default is SYS1.CBC.**

## Examples

None.

## References

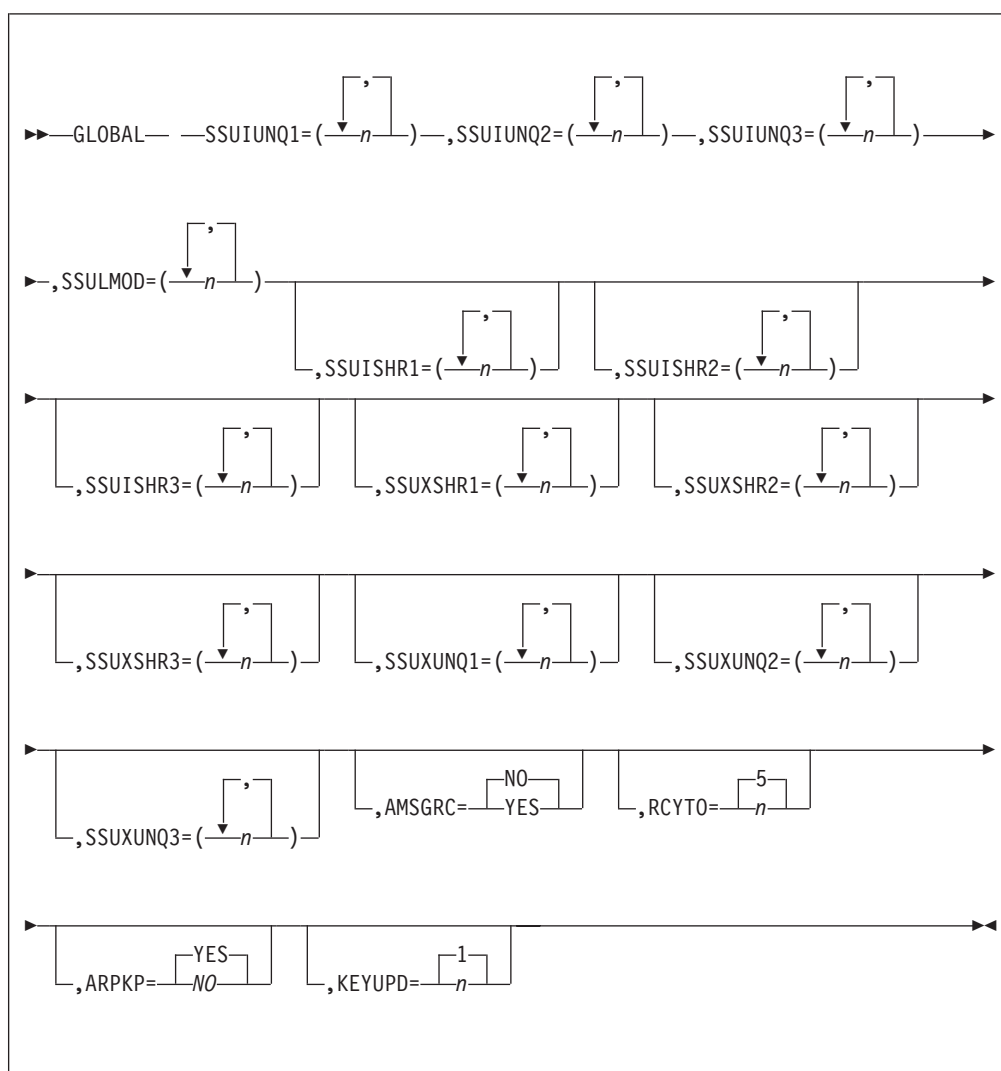
None.

## GLOBAL

The GLOBAL macro is used to specify the required size of the main storage global areas. You should refer to *TPF System Installation Support Reference* for information about global areas before providing inputs to this macro.

The GLOBAL macro is required.

## Format



### Notes:

1. When the GLOBAL macro is coded in a non-MDBF (multiple database function) environment, the SSU... parameters can only be coded with one value each. For an MDBF user, up to 94 individual values may be coded. The number of values specified should be equal to the number of subsystem users defined in the SSDEF macro.

**Note:** The maximum number of subsystem users per subsystem is 94. The maximum number of subsystem users in the entire system is 126. The total number of subsystems plus subsystem users must be less than or equal to 128.



## GLOBAL

2. Due to the limitation of the macro language, only 255 characters can be input on a parameter. Therefore, if the total subsystem user global definitions exceed 255 characters (including syntax), it will be necessary to code succeeding GLOBAL macro statements.
3. If **either** the SSUXSHRx or SSUXUNQx parameters is coded with a non-zero value then an extended global area will be defined in upper main storage just below the CIO area. This area will be equal in size to the sum of the total specified in all SSUXSHRx parameters, plus the number of I-streams times the total specified in all SSUXUNQx parameters.
4. A Global Attribute Table (GAT) will always be generated. The main storage address portion of each global directory entry will always consist of 4 bytes.

### **SSUIUNQx=(n1-n94)**

Specifies the maximum number of 4K-byte blocks required to contain the I-stream unique primary globals and global directories for a given global area x, where x = 1, 2 or 3. All values coded for SSUIUNQx must be nonzero. The size of each unique global area will be the same on all I-streams. The valid range is 1–65 535.

### **SSULMOD=(n1-n94)**

Specifies the core load mode number for each subsystem user to be stored in keypoint 'A'. (Refer to notes above.)

The load mode specified in program test vehicle (PTV) test units overrides this value.

These numbers are treated as hexadecimal values, and the valid range is X'0'–X'FF'. See *TPF System Installation Support Reference* for more information about core load mode numbers.

### **SSUISHRx=(n1-n94)**

Specifies the number of 4K-byte blocks required to contain the I-stream shared primary global records for global area x, where x = 1, 2 or 3. The default value is zero. The valid range is 1–65 535.

**Note:** For the first or only subsystem user in the TPF system (the BSS), the value coded for SSUISHR3 must specify enough storage to contain the RCAT. In addition, if SLC support is genned in the system, the value coded for SSUISHR2 must specify enough storage to contain the special program records. Refer to the *TPF System Installation Support Reference* for more information.

### **SSUXSHRx=(n1-n94)**

Specifies the number of 4K-byte blocks required to contain the I-stream shared extended global records for global area x, where x = 1, 2 or 3. The default value is zero. The value(s) specified will be added to the values coded for SSUXUNQx to calculate the size of the extended global areas reserved for the main I-stream's extended globals. The valid range is 1–65 535.

### **SSUXUNQx=(n1-n94)**

Specifies the maximum number of 4K-byte blocks required to contain the I-stream unique extended globals for a given global area x, where x = 1, 2 or 3. The default value is zero. The value(s) specified will be used as the size of the extended global areas reserved for the application I-streams. The valid range is 1–65 535.



**AMSGRC=NO|YES**

Application message recovery is to be supported. See the *TPF Data Communications Services Reference* for more information about application recovery.

**RCYTO=5|n**

Application message recovery CRET timeout value in seconds, where n=any valid number from 1 to 255. This parameter is not valid if AMSGRC=NO. See the *TPF Data Communications Services Reference* for more information about application recovery.

**ARPKP=YES|NO**

Defines if application recovery package keypointing is to take place. See the *TPF Data Communications Services Reference* for more information about application recovery.

**KEYUPD=1|n**

Specifies the time for controlling the frequency of global keypoint updating. Time is specified in second increments with a minimum of 1 second and a maximum of 60 seconds.

## Examples

**TPF system with MDBF support:** The following specifies the size of the three global areas in memory for a system with two subsystem users. Both subsystem users have the same size I-stream unique primary areas (6,6; 2,2; 2,2), but the second subsystem user requires less storage for its I-stream shared records (1,0,1), and has no shared global 2 records. The two subsystems have different load modes (8,9). In addition, the same I-stream unique extended global areas are defined (5,5; 1,1; 3,3), but the second subsystem has no I-stream shared extended globals at all. Application message recovery is not required.

```
GLOBAL      SSUI SHR1=(4,1),SSUI SHR2=(3,0),SSUI SHR3=(3,1),      x
             SSUI UNQ1=(6,6),SSUI UNQ2=(2,2),SSUI UNQ3=(2,2),      x
             SSUX SHR1=(3,0),SSUX SHR2=(1,0),SSUX SHR3=(1,0),      x
             SSUX UNQ1=(5,5),SSUX UNQ2=(1,1),SSUX UNQ3=(3,3),      x
             SSULMOD=(8,9)
```

**TPF system without MDBF support:** The following specifies the size of the I-stream shared and I-stream unique areas for global areas 1, 2, and 3, and the core load mode number. The total size of the I-stream shared area below 16M will be 3 + 5 + 4 = 12 4K-byte blocks, and the size of the I-stream unique area will be 6 + 10 + 8 = 24 4K-byte blocks. Because all extended global parameters are omitted, the system will be generated without extended globals, but a GAT will still be created. Application message recovery will be supported with the RCYTO and ARPKP parameters defaulting to 5 seconds, and YES, respectively. The global keypoints will be updated every 3 seconds.

```
GLOBAL      SSUI SHR1=3,SSUI SHR2=5,SSUI SHR3=4 ,                x
             SSUI UNQ1=6,SSUI UNQ2=10,SSUI UNQ3=8,                x
             SSULMOD=7,AMSGRC=YES,KEYUPD=3
```

## References

- *TPF System Installation Support Reference*
- *TPF Data Communications Services Reference*.

## GLSYNC

The GLSYNC macro is used to specify the synchronized globals, both fields and records, needed by the various system users.

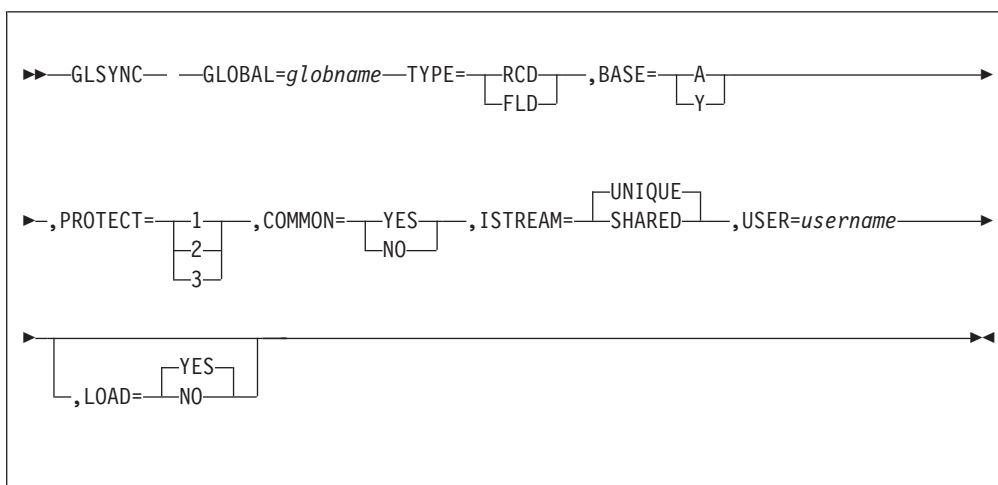
One GLSYNC macro is required each time a subsystem user seeks to synchronize a particular global field or record.

This macro will generate the System Interprocessor Global Table (SIGT) program and the equate list, SYNLST. SIGT requires that space be reserved on DASD for the #SGFRI record type. For each subsystem user, one #SGFRI-type record is required for each global field that is to be synchronized. When no GLSYNC macros are coded, at least two #SGFRI records are still needed. The #SGFRI record type is defined by coding the SIP RAMFIL macro. (See “RAMFIL” on page 290 for additional information.)

The first time the macro is coded for a global, all parameters must be specified. Thereafter any invocation of the macro for that particular global may be coded with only the global name, the user and the common value. If the other parameters are coded, they **must not differ** from the original invocation.

This macro is not required.

## Format



### GLOBAL=*globname*

A 1–8-character name to be assigned to the global. The name must be unique among the globals of either type.

**Note:** @SYNCD and @RCOUP are reserved for IBM.

### TYPE

Specify one of the following:

**RCD** The global is a synchronized global record

**FLD** The global is a synchronized global field

When TYP=FLD and BASE=Y, COMMON must equal YES. When TYP=FLD and BASE=A, COMMON must equal NO.

**BASE**

Specify one of the following:

- A**      The global directory item is in GLOBA.
- Y**      The global directory item is in GLOBY.

**PROTECT**

Specify one of the following:

- 1**      The global resides in GLOBAL1
- 2**      The global resides in GLOBAL2
- 3**      The global resides in GLOBAL3

Only global areas 1 and 3 are protected.

**COMMON**

Specify one of the following:

- YES**    The global is common among subsystem users.
- NO**     The global is unique to each user.

If any subsystem user has a global with COMMON=YES specified, then the primary subsystem user (first argument of SSUID of SSDEF) must have COMMON=YES specified for the same global.

If COMMON is not specified, it is set to NO unless TYPE=FLD and BASE=Y, then it is set to YES.

**ISTREAM**

Specify one of the following:

- UNIQUE**      The global is unique to each I-stream.
- SHARED**      The global is shared by all I-streams. (Applications must serialize access for shared globals.)

**USER=*username***

The 1–4-character name of the subsystem user. This name must be present within the SSUID parameters of SSDEF. If SSDEF is not coded, allow USER to default. If USER is not coded, it will default to the first SSUID1 defined in the SSDEF macro.

**LOAD**

This parameter is only to be coded when TYPE=FLD. It will be ignored otherwise.

- YES**    Specifies that the global field is to loaded.
- NO**     Specifies that the global field is not to be loaded.

When a SIGT created from a GLSYNC macro is loaded from the loader general file to an online system for each field with LOAD=YES, the data used by that system for the synchronized field is taken from the global record. If LOAD=NO, then the current value of the field remains.

This option gives the user the ability to alter the SIGT without altering the value of some or all of the synchronized global fields. By use of pilot tapes and coding LOAD=YES, the user can selectively alter synchronized global fields by loading new data to the global record holding the field. Fields that will not be altered should be coded LOAD=NO.

## GLSYNC

### Examples

The following example is presumed to take place in a loosely coupled environment, in which SSU1, SSU2, and SSU3 are defined as subsystem users in SSDEF.

The global record, MYGLOB, which is a member of GLOBA and GLOBAL1 will occur twice, once for SSU1 and once for SSU2.

The global field, MYFLD, which is a member of GLOBY and GLOBAL3 will occur only once, shared by SSU1 and SSU3.

cc10	cc17	cc72
GLSYNC	GLOBAL=MYGLOB,TYPE=RCD,PROTECT=1,BASE=A,COMMON=NO, USER=SSU1,ISTREAM=SHARED	X
GLSYNC	GLOBAL=MYGLOB,COMMON=NO,USER=SSU2	
GLSYNC	GLOBAL=MYFLD,TYPE=FLD,PROTECT=3,BASE=Y,COMMON=YES, USER=SSU1,ISTREAM=UNIQUE	X
GLSYNC	GLOBAL=MYFLD,COMMON=YES,USER=SSU3,ISTREAM=UNIQUE	

### References

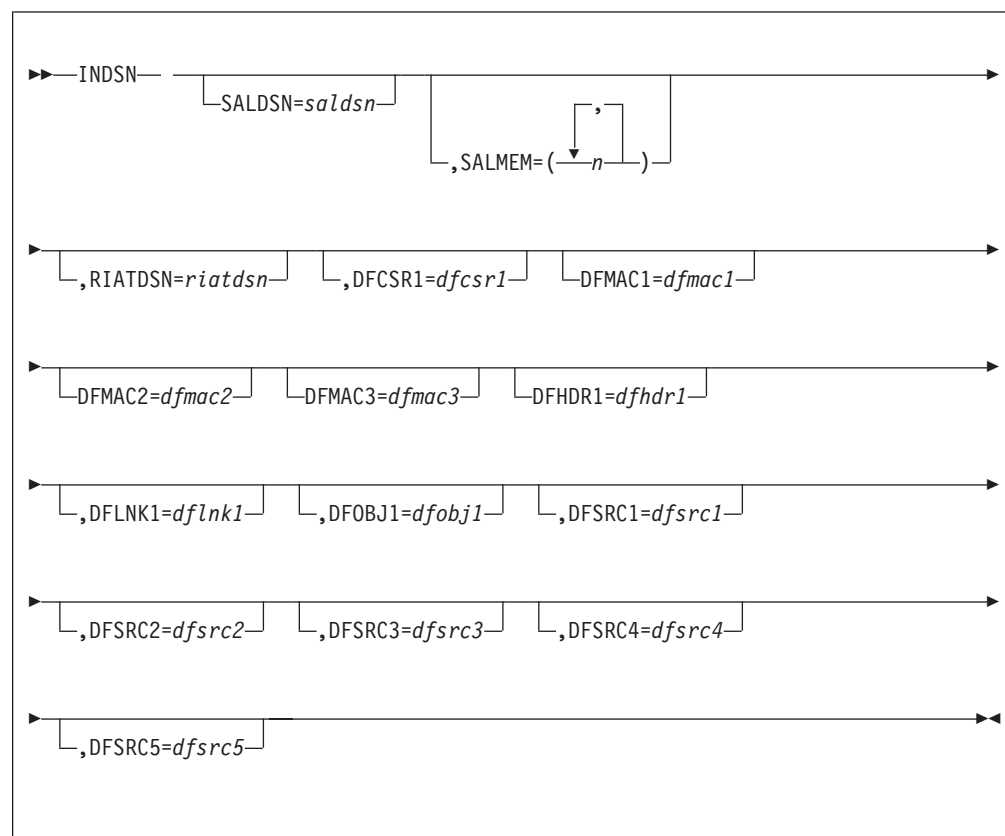
SYNCC in *TPF General Macros*.

## INDSN

The INDSN macro provides data set names and members referenced in SIP Stage II.

This macro is optional; if it is not coded, default parameters are provided.

## Format



### **SALDSN=saldsn**

The name of the data set that contains the input decks for the system allocator, SALO. IBMPAL will be taken from ACP.SALIN.RELnn regardless of what is coded for this parameter.

**Attention:** The data set referred to by this parameter must have the same file attributes (RECFM, block size and LRECL) as the data set in which IBMPAL resides.

**The default is ACP.SALIN.RELnn, where nn refers to the release version number.**

### **SALMEM=(n1,n2,...n15)**

Up to 15 user-coded decks that are concatenated with the appropriate system input deck (IBMPAL) and used as input to the allocator. Do not code the name IBMPAL because this deck is automatically included. Names must contain 1 to 6 characters.

**There are no defaults for this parameter.**

### **RIATDSN=riatdsn**

The name of the data set that contains the RIAT to be assembled.

*The default is ACP.SRCE.OL.RELnn, where nn refers to the release version number.*

#### **DFCSR1=dfcsr1**

The name of the first data set that contains the TPFDF C source code.

*The default is BDF.V1R1M3.BDFCSR1 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid.*

#### **DFMAC1=dfmac1**

The name of the first data set that contains TPFDF assembler macros.

*The default is BDF.V1R1M3.BDFMAC1 for TPF system with TPFDF installed. If TPFDF is not installed, do not code this parameter because it is not valid.*

#### **DFMAC2=dfmac2**

The name of the second data set that contains TPFDF assembler macros.

*The default is BDF.V1R1M3.BDFMAC2 for TPF system with TPFDF installed. If TPFDF is not installed, do not code this parameter because it is not valid. Specify DFMAC2=NONE if there is no second dataset.*

#### **DFMAC3=dfmac3**

The name of the third data set that contains TPFDF assembler macros.

*The default is BDF.V1R1M3.BDFMAC3 for TPF system with TPFDF installed. If TPFDF is not installed, do not code this parameter because it is not valid. Specify DFMAC3=NONE if there is no third dataset.*

#### **DFHDR1=dfhdr1**

The name of the data set that contains TPFDF C language header files.

*The default is BDF.V1R1M3.BDFHDR1 for systems with TPFDF installed. If TPFDF is not installed, do not code this parameter because it is not valid.*

#### **DFLNK1=dflnk1**

The name of the first data set that contains TPFDF link-edit code.

*The default is BDF.V1R1M3.BDFLNK1 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid.*

#### **DFOBJ1=dfobj1**

The name of the first data set that contains TPFDF object code.

*The default is BDF.V1R1M3.BDFOBJ1.yyyy, for a TPF system with the TPFDF product installed, where yyyy is the subsystem name. If the TPFDF product is not installed, do not code this parameter because it is not valid.*

#### **DFSRC1=dfsrc1**

The name of the first data set that contains the TPFDF source code.

*The default is BDF.V1R1M3.BDFSRC1 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid.*

#### **DFSRC2=dfsrc2**

The name of the second data set that contains the TPFDF source code.

*The default is BDF.V1R1M3.BDFSRC2 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid. Specify DFSRC2=NONE if there is no second dataset.*

#### **DFSRC3=dfs3**

The name of the third data set that contains the TPFDF source code.

*The default is BDF.V1R1M3.BDFSRC3 for a TPF system with the TPFDF product installed. If TPFDF is not installed, do not code this parameter because it is not valid. Specify DFSRC3=NONE if there is no third dataset.*

#### **DFSRC4=dfs4**

The name of the fourth data set that contains the TPFDF source code.

*The default is BDF.V1R1M3.BDFSRC4 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid. Specify DFSRC4=NONE if there is no fourth dataset.*

#### **DFSRC5=dfs5**

The name of the fifth data set that contains the TPFDF source code.

*The default is BDF.V1R1M3.BDFSRC5 for a TPF system with the TPFDF product installed. If the TPFDF product is not installed, do not code this parameter because it is not valid. Specify DFSRC5=NONE if there is no fifth dataset.*

## Examples

None.

## References

See "CONFIG" on page 219 for more information about how to find out whether TPFDF is installed.

## IODEV

The IODEV macro is used to specify the characteristics and system requirements for specific input/output (I/O) devices. Each disk, 37x5 communication control unit, 3088 addresses used by SNA CTC (DVTYP=SCTC), or tape I/O device which is uniquely addressable must be specified with an IODEV macro.

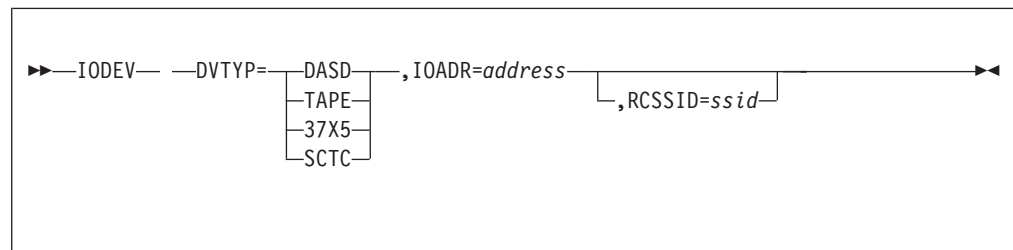
In a multiple database function (MDBF) environment, all subsystems use the highest and lowest channel addresses calculated from the basic subsystem.

For 37x5 communication control units and SNA channel-to-channel (CTC) connections, only one symbolic device address can be coded per use of the IODEV macro. The total number of 37x5 communication control units and SNA CTC connections defined by the IODEV macro cannot exceed 255. This information is used to build the Symbolic Device Address Table (SDAT).

A symbolic device address associated with a 37x5 communication control unit running EP only or with the EP portion of a communication control unit running PEP must not be specified using the IODEV macro. Instead, the NETWK macro is used to specify 37x5 communication control units running EP.

For tape devices, an entry is created in the Tape Control Unit Cross Reference Table (COSY) for each IODEV coded. This table is used for validation during restart and tape reconfiguration. Each logical channel/control unit value specified defines a tape subsystem; up to 16 devices may be configured on a subsystem. It is recommended that sufficient tape subsystems are generated to allow for expansion without the need for regeneration. Tape drives are added online, and the results are keypointed (CTK0, CTK1, and CTKB), and thus preserved across a system IPL.

## Format



### DVTYP

Specify one of the following:

**DASD** Specifies a series of eight devices:

- 3350 Model 2 or Model 2F
- 3375 Model 1 or Model 1F
- Any 3380 model (including 3390s in 3380 emulation mode)
- Any 3390 model.
- Any 9345 model.

**TAPE** Specifies a series of 16 tape devices.

**37x5** Specifies a single IBM communication control unit supported by TPF.

**SCTC** Specifies a single SNA channel-to-channel (CTC) connection.



**IOADR=address**

This parameter specifies the symbolic device address for each type of device.

For DASD devices, this address will encompass eight units/spindles. The symbolic device address must consist of four hexadecimal digits, must be in the range X'0100' to X'7FF8', and the last digit must be either a zero or an eight. The first two digits represent the logical channel number where the device is attached. The third and fourth digits represent the logical control unit, the string address, and the starting device number. For more information and examples refer to "Database Device Addressing" on page 79.

For tape devices, IOADR must consist of two hexadecimal digits (X'10' to X'FF') that represent the logical channel/control unit subsystem.

Note that the device numbers specified in the source for the IOCP represent symbolic (logical channel/control-unit/device) addresses to TPF.

A channel address of 0 should not be used for DASD or tape devices.

For 37x5 communication control units or SNA CTC connections this address defines one specific symbolic device address (SDA). The symbolic device address must be a four-digit hexadecimal value in the range X'0001' through X'7FFF'.

**RCSSID=ssid**

This parameter specifies a record cache subsystem SSID (RCSSID) associated with a DASD string attached to a 3990 Record Cache Subsystem Control Unit. This value is used to generate the maximum record cache SSID value configured as part of keypoint 0 to be used in allocating system storage for the record cache subsystem status table (SSST).

Code this parameter only when DVTYPE=DASD and when this DASD string is serviced by a 3990 Record Cache Subsystem Control Unit. If DVTYPE is not equal to 'DASD', RCSSID is ignored.

RCSSID accepts any four-digit hexadecimal number in the range X'0001' to X'FFFF', inclusive. If you fail to code RCSSID for 3990 cached DASD strings, an improper record cache subsystem status table may be created at load time.

**Note:** Although a full 16-bit hexadecimal SSID value is accepted, there is a maximum limit on the total number of RCS subsystems (6, 630 or X'19E6') that can be defined. This limit is imposed by the number of fixed file records that are allocated for storing SSST information.

## Examples

**Example 1:** The following specifies a DASD string with eight devices connected on address 01A0.

```
IODEV  DVTYPE=DASD,IOADR=01A0    /* UNITS 01A0-01A7 */
```

**Example 2:** The following specifies a DASD string with 16 devices connected on address 01A0.

```
IODEV  DVTYPE=DASD,IOADR=01A0    /* UNITS 01A0-01A7 */
IODEV  DVTYPE=DASD,IOADR=01A8    /* UNITS 01A8-01AF */
```

**Example 3:** The following specifies a TAPE subsystem that defines logical addresses 180–18F.

```
IODEV  DVTYPE=TAPE,IOADR=18
```

**Note:** No other devices can be specified with a logical address starting with 18.

## IODEV

**Example 4:** The following specifies a 37x5 communication control unit with an address of X'1C' and an SNA CTC connection on symbolic address X'752E'.

```
IODEV DVTYP=37X5,IOADR=001C  
IODEV DVTYP=SCTC,IOADR=752E
```

**Note:** The IOADR parameter must be four hexadecimal digits for a 37x5 or SNA CTC in the range X'0001' through X'7FFF', and padded on the left with zero(s).

## References

None.

## LINES

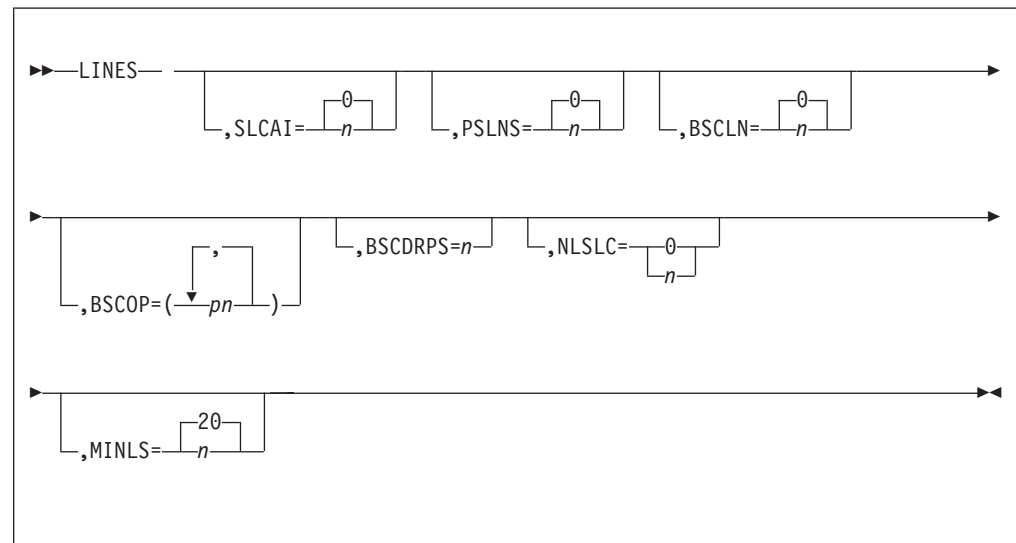
The LINES macro is used to specify the numbers and types of non-SNA communication lines to be attached to the system or directly to the channel for local 3272s.

The number of lines defined in the LINES macro must agree with the number specified during non-SNA communications generation. See “Non-SNA Communications” on page 335. You should also refer to the LINEQ macro listing before coding this macro.

If the system being generated consists of only SNA communications lines, then you do not need to code this macro; SIP will automatically assign the default values shown below.

The LINES macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

## Format



**Note:** The following parameters are only used for synchronous line control (SLC) lines. If entered, N2703 in NETWK must not equal zero.

**SLCAI=0|n**

Number of synchronous link control AI lines attached to the EP communication controllers. If entered, SYNCLK macro must be coded.

**PSLNS=0|n**

Number of SLC pseudo ALC lines to be assigned to the system. If this field is entered, NLINK parameter in SYNCLK must not equal zero.

**Note:** The following parameters are only used for BSC lines. If entered, N2703 in NETWK must not equal zero.

**BSCLN=0|n**

Number of bisynchronous lines to be attached to the EP communication controllers.

## LINES

### **BSCOP=(p1, p2,...pn)**

This optional keyword specifies options for BSC lines. The parameter list may be coded in any order and each entry must be separated by a comma. The valid BSCOP parameters (p1,p2,...pn) are:

- CT** The system will be a BSC control station
- EB** BSC EBCDIC code will be used
- MP** BSC multipoint lines will be included
- PP** BSC point to point lines will be included
- US** BSC USASCII code will be used

### **BSCDRPS=n**

The total number of BSC drops on the system. A BSC drop is defined as the following:

1. A point-to-point BSC line connecting a processor to a processor is defined as 1 drop.
2. A multipoint BSC line with two processor stations attached to it is defined as 2 drops.
3. A point-to-point BSC line connecting a processor to a processor station is defined as 1 drop.

#### ***Default is as follows:***

1. If BSCOP=MP is coded then the number of BSC drops will equal the total number of BBSAT macros coded multiplied by the total number of BSC lines (BSCLN).
2. If BSCOP=PP is coded and not MP then the total number of BSC drops will default to the number of BSC lines coded in the BSCLN parameter.

**Note:** The NLSLC parameter is only used for IBM 3270 local lines. If entered, the D3270 parameter in NETWK must not be NO.

### **NLSLC=0|n**

Number of 3270 local lines.

**Note:** The MINLS parameter applies to all of the LINES macro parameters.

### **MINLS=20|n**

A valid hexadecimal number representing the first symbolic line number. For more information about how MINLS is used, see the LNSY parameter description of the SKLNG macro in "SKLNG" on page 338.

## Examples

A system contains 9 BSC lines and there are a total of 35 point-to-point and multipoint drops on the BSC lines. The LINES macro can be coded as follows:

```
LINES  BSCLN=9,BSCOP=(MP,PP,CT,EB,US),  X
      BSCDRPS=35
```

## References

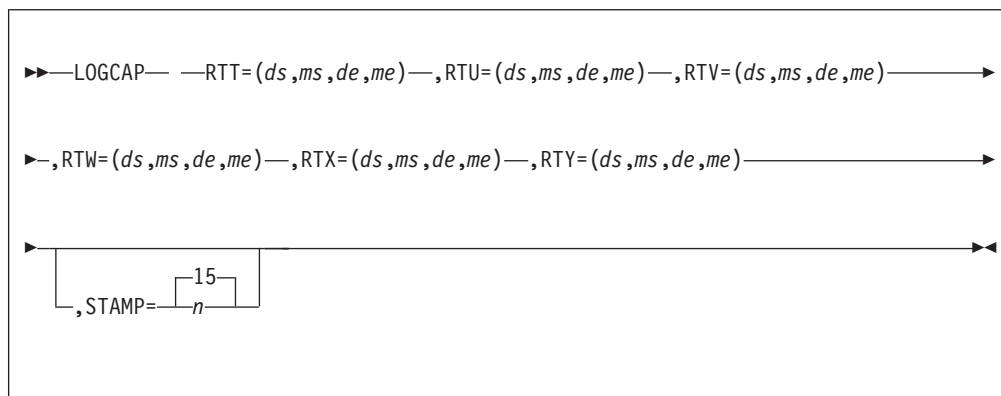
None.

## LOGCAP

The LOGCAP macro is used to specify information relating to LOG/XCP processing. Values are entered to specify the online module ranges to be assigned to each of the LOG/XCP exception recording tapes. LOGCAP is required if the capture/restore facility is desired. Detailed explanations of how to initialize these fields are available in “File Capture and Restore” in the *TPF Database Reference*.

LOGCAP is required if the capture/restore facility is desired.

## Format



**Note:** Parameters RTT, RTU, RTV, RTW, RTX and RTY are identical in definition of contents. Therefore only one definition is provided.

At least one of these parameters must be defined. It does not matter which one is supplied.

### RTa

Where a = T, U, V, W, X or Y, the tape name.

- ds** The starting DASD device type for the tape. DEVA for device A, DEVB for device B, DEVC for device C, or DEVD for device D. (See the ONLFIL macro.)
- ms** The starting relative module number within the starting device type (i.e., 0 is the first relative module number, the number of modules minus one is the last module number).
- de** The ending DASD device type for the tape (refer to ds for parameter definition).
- me** The ending relative module number within the ending device type (refer to ms for parameter definition).

This range defines the valid disk low and high contiguous module numbers assigned to this tape for LOG/XCP capture/restore.

Tapes must cover all disk module numbers on the system without overlaps. For example, a system with 4 device A and 2 device B modules might be assigned:

```

RTV = (DEVA,0,DEVA,1)
RTU = (DEVA,2,DEVA,3)
RTY = (DEVB,0,DEVB,0)
RTW = (DEVB,1,DEVB,1)

```

## LOGCAP

**Note:** The starting (ms) and ending (me) ranges will be cross-checked against the inputs provided to the ONLFIL macro. If the logging ranges exceed the module ranges defined by ONLFIL, then an error or warning message is issued. The module ranges cannot exceed the maximum number allowed, 3998, regardless.

**STAMP=15|n**

Time interval, in minutes, at which a time stamp will be written to all logging tapes. Zero is not valid.

## Examples

LOG/XCP recording is to be done to tape name RTV for device A. This device has two modules. All other values can default. Code the LOGCAP macro as follows:

```
LOGCAP RTV=(DEVA,0,DEVA,1)
```

Possible errors in assigning disk modules with the same system might be:

```
RTT = (DEVA,1,DEVA,3)
```

```
RTY = (DEVB,0,DEVB,1) all modules not included
```

or,

```
RTY = (DEVA,0,DEVA,2) RTY and RTW overlap. They both attempt
```

```
RTW = (DEVA,2,DEVA,3) to map DEVA 2.
```

```
RTY = (DEVB,0,DEVB,1)
```

## References

None.

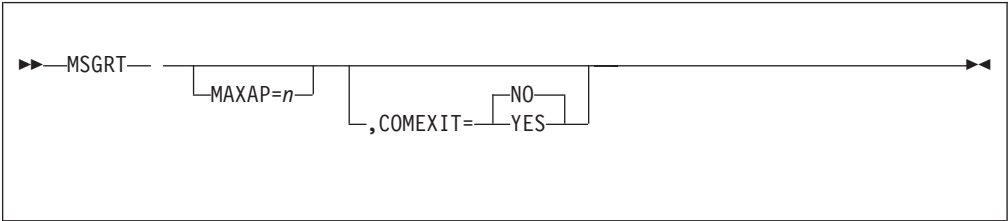
MSGRT

The MSGRT macro is required to specify message router availability and set message router switches and parameters which apply only to the processor being generated.

MSGRT must be coded, although all the parameters are optional.

In TPF systems containing the multiple database function (MDBF), message router support is required only for the basic subsystem generation and is not to be included when generating any other type of subsystem.

Format



MAXAP=n

Specifies the maximum number of user applications that the user is ever expected to define.

The valid range is 1 to 256-(m+s). Default is 256-(m+s).

m      The number of fixed system applications, which is equal to:  
         2 × (number of systems in the network) + 1.

s      The number of optional system applications, equal to 1 if airline's reservation package is specified via the RES parameter of the MSGRTA macro. Otherwise, equal to 0.

**Note:** The number of entries created in the RCAT table is based on the total number of applications. This includes the user applications specified by the MAXAP parameter, plus the number of fixed and optional system applications as defined above.

COMEXIT=NO|YES

Specifies whether or not a user-written routine will be invoked for every input message processed by the message router package.

**Note:** Inclusion of this macro causes the unsolicited message package to be included in the system.

Examples

None.

References

Data macro RC0AT.

---

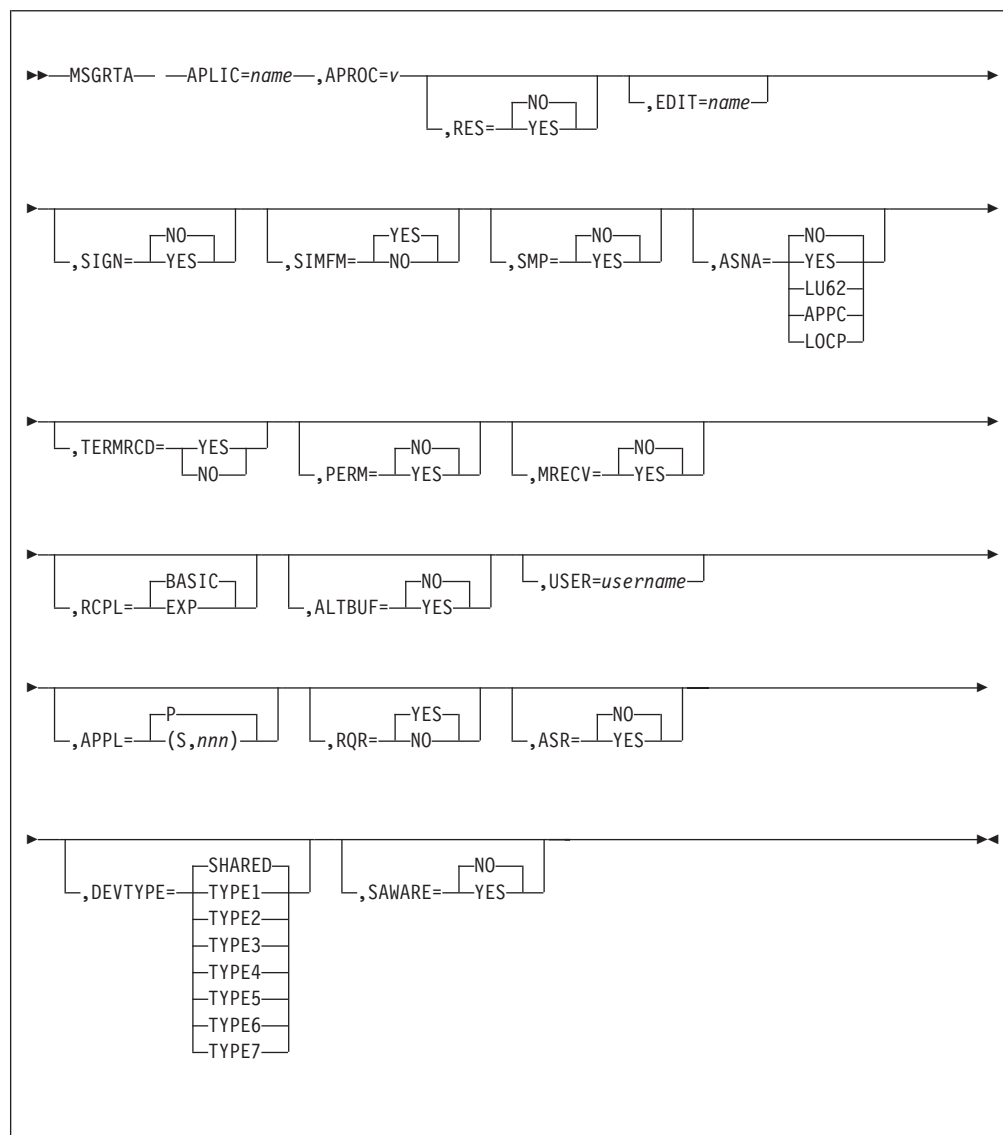
**MSGRTA**

The MSGRTA macro is used to specify user application programs that exist in the host processors of the network. The data supplied via this macro is used to create the RCAT initialization table (RCIT) and is also used in the creation of the SNA table via the offline SNA table generation program. All applications that are resident in the processor being generated must be specified. Applications that use the TPF message routing protocol (that is, routing control parameter list (RCPL) in the message header) and are not resident in the generated processor, but may receive messages routed from the generated processor must also be specified. In addition, the MSGRTA macro will identify the TPF processor IDs of attached processors to the TPF complex being generated.

The MSGRTA macro is only to be coded when generating the basic subsystem.



## Format



### APLIC=*name*

Application name consisting of four alphanumeric characters. The first three characters may not be TPF, CLG, LOG, NEF, or SMP.

Entries in the RCAT initialization table are automatically generated for the following fixed and optional system applications:

**CLG***x* Where *x* is a TPF processor ID

**LOGI** Log processor

**SMP***x* Where *x* is a TPF processor ID

SIP supports up to 256 message router application names (user plus system names).

To use TPF/APPC support in a loosely coupled complex, you must define one TPF/APPC service LU for every processor in the complex. For these definitions specify:

## MSGRTA

APLIC=SVCx, where x is the processor ID  
EDIT=CHDD  
ASNA=APPC  
APROC=x, where x is the processor ID.

**Attention:** An application name must be unique. Only one MSGRTA statement may be coded per application name.

### APROC=v

One character to identify the processor in which the application resides. For applications resident in the generated processor, this must be the TPF processor ID defined in the SYSID parameter of the CONFIG macro.

The TPF processor ID specified here will either identify a processor in the SYSID parameter of CONFIG or define a processor considered to be attached to the TPF complex being generated.

In a loosely-coupled system an asterisk (\*) should be coded as the processor due to recovery considerations. The asterisk (\*) indicates that the application is present in all processors.

The character coded for this parameter must be in the range A–Z or 0–9.

### RES

Specify one of the following:

**NO** This application is not a RES0 type application.

**YES** This application is a RES0 type application. RES0 will no longer be included automatically by the system.

If RES=YES is coded, the other parameters should be coded as follows:

EDIT=UII1  
SIGN=YES  
SIMFM=YES

**Note:** Any general reservation type application which requires the use of AAA records must code this parameter as YES. This parameter is not necessarily connected with the CONFIG RES= parameter.

### EDIT=name

The four character name of the program that is to receive all messages destined for this application. This parameter is required only for applications which are resident in the generated processor. The first character must be alphabetic, the remaining characters are alphanumeric. The default name of CVVC will be assigned if this parameter is omitted for an application that is resident in the generated processor. By assigning this default name, the SIP Stage I process can be allowed to complete. The user, if desired, may then modify the Stage I output for program COHA/B (RCIT) to contain the desired program name or rerun Stage I with the corrected MSGRTA macro.

To use the TPF/APPC package, specify:

EDIT=CHDD  
ASNA=APPC, ASNA=LU62, or ASNA=LOCP  
DEVTYPE=SHARED

**SIGN**

Specify one of the following:

- NO** No sign-in required.
- YES** A sign-in message is required by the application.

**SIMFM**

Specify one of the following:

- YES** A simulated 4505 type terminal message format will be used for this application for messages to or from a 3270 type terminal.
- NO** A user format for messages.

**SMP**

Specify one of the following:

- NO** The application is not an SMP type of application.
- YES** The application is an SMP type of application. If specified, the other parameters should be coded as follows:
  - ASNA=NO
  - MRECV=NO
  - PERM=YES
  - RAPPL=NO
  - RCPL=BASIC
  - SIGN=NO
  - SIMFM=YES
  - APPL=P
  - ALTBUF=NO

**ASNA**

Specify one of the following:

- NO** The application will address all of its terminals (logical units) using the LNIATA type of terminal address. A terminal control record (RCB or AAA) will be retrieved and passed to the application program with input messages.
- YES** The application program will address all of its terminals (logical units) using only the SNA resource identifier (RID). Also, no terminal control record (RCB or AAA) is to be passed to the application with input messages.
- LU62** The application will communicate as a TPF/APPC LU. You must specify EDIT=CHDD, RCPL=EXP, and DEVTYPE=SHARED when using this option.

**Note:** It is not necessary to define remote LU62 nodes here. You can define these using the RSC statement in the offline ACF/SNA table generation program (OSTG). See the *TPF ACF/SNA Network Generation* for details.

- APPC** The application communicates as the default local TPF/APPC LU or a TPF/APPC service LU. You must specify EDIT=CHDD, RCPL=EXP, and DEVTYPE=SHARED when using this option.
- LOCP** The application communicates as the local APPN control point (CP).

## MSGRTA

You must specify EDIT=CHDD, RCPL=EXP, APROC=\*, and DEVTYPE=SHARED when using this option. Only 1 local APPN CP can be defined.

**Note:** This parameter must be coded as YES, LU62, or APPC when the APPL parameter is coded as S.

### TERMRCD

Specify one of the following:

**YES** The terminal record will be returned to the application. This option may be specified only for non-SNA applications (ASNA=NO). *If ASNA=NO, **TERMRCD=YES is the default.***

**NO** The terminal record will not be returned to the application. *If ASNA=YES, **TERMRCD=NO is the default.***

### PERM

Specify one of the following:

**NO** This application will not be permanent.

**YES** This application will be a permanent application when functionally added to the RCAT (it cannot be deleted).

### MRECV=NO|YES

This parameter defines whether this application supports SNA system message recovery.

### RCPL=BASIC|EXP

This parameter defines whether this application uses the basic form of the RCPL or the expanded format.

### ALTBUF

Defines the buffer size to be used for the application using the extended 3270 devices.

**NO** Default buffer size for the extended 3270 LU.

**YES** Alternate buffer size for the extended 3270 LU.

### USER=username

This parameter is only to be coded in an MDBF system. One of the 1- to 4-character alphanumeric names used in the SSUID1-4 parameter of the SSDEF macro associates a particular subsystem user with an application. If omitted, the value defaults to the first/only name given in the SSUID parameter of the SSDEF macro.

If the system is an MDBF system, all applications must be specified in the basic subsystem. The default user is the first/only name given in the SSUID parameter of the basic subsystem generation.

Any users appearing in MSGRTA statements must appear in some subsystem's SSUID parameter (SSDEF).

**If SSDEF is not specified, USER defaults to the value SSU0.**

**Attention:** An application may only be specified for one user. (See APLIC parameter above.)

### APPL=P|S,nnn|S,1

This parameter indicates whether the application can act as both a primary LU (PLU) and a secondary LU (SLU), or as only a PLU.

- P** Defines this application as a primary logical unit only. This is the default.
- S** Defines this application as a secondary logical unit, where *nnn* is a number defining the concurrent session limit that this SLU can participate in with a single PLU in another domain. A maximum of 255 can be specified. The default is 1.

The SLU name is created by concatenating the 4-character PLU name with the CPU ID specified on the APROC parameter and with a 3-digit decimal number.

**Note:** If this parameter is coded as S, the ASNA parameter must be coded as YES, LU62, or APPC.

### **RQR=NO|YES**

This parameter defines whether the SNA command RQR is being supported for SLU-P threads.

**Note:** If this parameter is coded as YES, ASNA must be YES also.

### **ASR**

Specify one of the following:

- NO** SLU-P threads will not be resynchronized during restart.
- YES** TPF will resynchronize SLU-P threads during restart.

**Note:** If this parameter is coded as YES, ASNA must be YES also.

### **DEVTYPE**

This parameter allows you to specify different message queues for different applications by defining which NCB slot (0–7) in an NCB directory record entry dynamic LU resources will use when they log on to the application. An NCB slot contains the address of a 381-byte long-term pool file NCB record. See *TPF ACF/SNA Data Communications Reference* for more information about dynamic LU resources, NCB directory records, and NCB slots.

**Note:** LU resources defined using the OSTG program use the same 381-byte fixed file NCB directory record, regardless of the application. Therefore, this parameter does not apply to LU resources that are defined using the OSTG program.

**SHARED** Specifies that a dynamic LU resource will use NCB slot 0 when it logs on to the application. Use this option if you want the application to share the same message queue with other applications.

**Note:** You must specify this option when you specify ASNA=LU62, APPC, or LOCP.

**TYPE1** Specifies that a dynamic LU resource will use NCB slot 1 when it logs on to the application. Use this option if you want the application to have a unique message queue.

**TYPE2** Specifies that a dynamic LU resource will use NCB slot 2 when it logs on to the application. Use this option if you want the application to have a unique message queue.

**TYPE3** Specifies that a dynamic LU resource will use NCB slot 3 when it logs on to the application. Use this option if you want the application to have a unique message queue.

## MSGRTA

- |              |   |
|--------------|---|
| <b>TYPE4</b> | Specifies that a dynamic LU resource will use NCB slot 4 when it logs on to the application. Use this option if you want an application to have a unique message queue. |
| <b>TYPE5</b> | Specifies that a dynamic LU resource will use NCB slot 5 when it logs on to the application. Use this option if you want an application to have a unique message queue. |
| <b>TYPE6</b> | Specifies that a dynamic LU resource will use NCB slot 6 when it logs on to the application. Use this option if you want an application to have a unique message queue. |
| <b>TYPE7</b> | Specifies that a dynamic LU resource will use NCB slot 7 when it logs on to the application. Use this option if you want an application to have a unique message queue. |

### **SAWARE=NO|YES**

This parameter defines whether secondary logical unit (SLU) threads will have session awareness support. See *TPF ACF/SNA Data Communications Reference* for more information about session awareness support.

## Examples

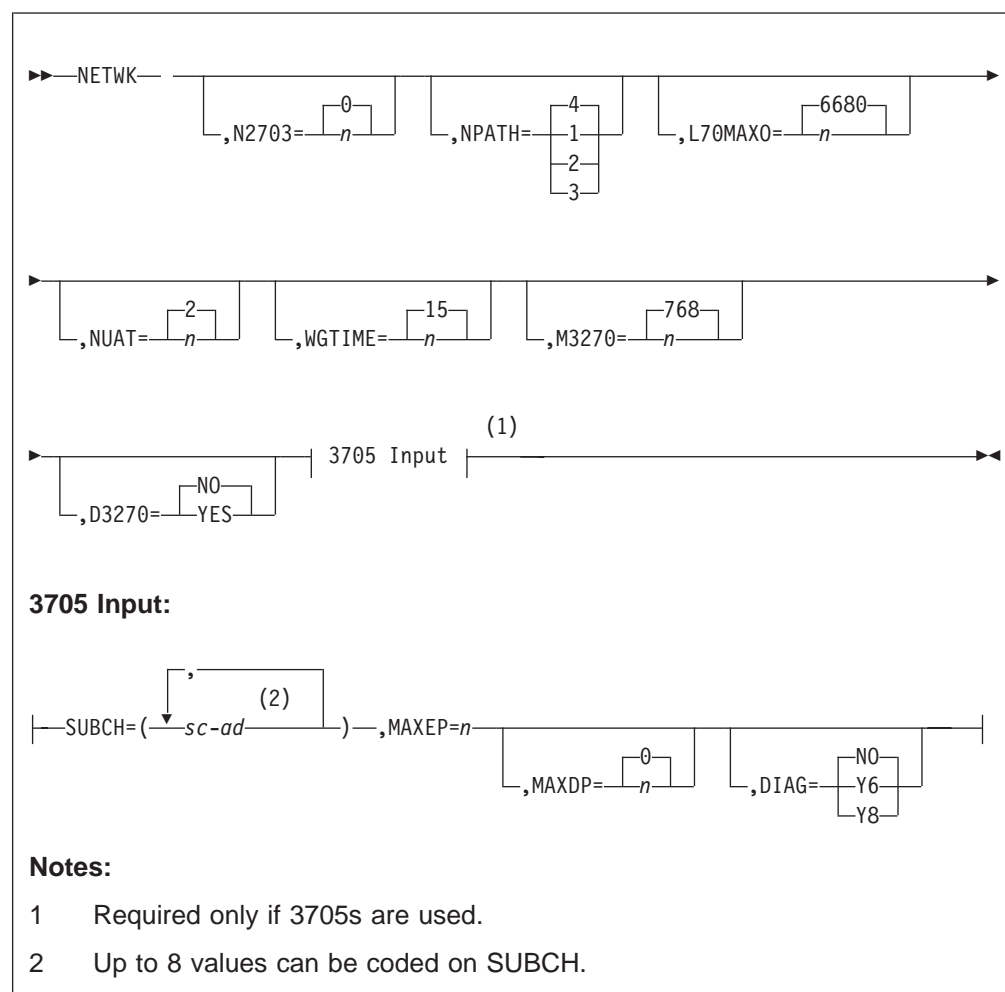
None.

## References

None.

The number of control units will be defined according to the number of subchannels. The control unit addresses will be in ascending order beginning with '00.'

## Format


$$N_{2703} = 0|n$$

Number of active EP TCUs that the system can recognize. Up to eight TCUs

## NETWK

may be entered. This parameter must be coded as nonzero (or the D3270 parameter must be coded as YES) if lines are coded on the LINES macro.

### **NPAT<sub>H</sub>=1|2|3|4**

Number of physical paths (primary and alternate) through which a symbolic line may be attached to the system. The maximum number is 4 (limited by system software).

### **L70MAXO=6680|n**

This parameter defines the size of the largest output message that can be sent to a 3270 local device. The value is numeric, and must be greater than 0.

**Default is 6680 (20 381-byte blocks).** Maximum is 6680.

**Note:** L70MAXO is used to compute the maximum number of message blocks per message. However, more message blocks may be required by the mapping user. Therefore, when using 3270 mapping, it is recommended that L70MAXO be set to its largest allowable values (that is, L70MAXO=6680).

### **NUAT=2|n**

The number of unique LNIATA's requiring an entry in the AAA initialization table (UAT). This value will be used to create the WGTA table and WGTA hash table. This value should reflect the user's future requirements for UAT expansion. The minimum value that can be coded is two times the number of processors defined in the CONFIG macro.

**Note:** If an automation gateway is connected to the system, additional UAT entries must be specified. The additional number of entries is the number of terminal addresses in the range 010003 to 0100mm, where mm is the maximum terminal address specified in your IBM Extended Operations Console Facility/2 configuration. (For additional information on the maximum terminal address, see the *IBM Extended Operations Console Facility/2 System Administrator's Guide*.)

### **WGTIME=15|n**

Specifies the time for controlling the frequency of keypointing the WGTA tables. The time specified is in minutes increments with the minimum time that can be specified being one minute and the maximum being 30 minutes.

**Note:** A RAMFIL statement for record type #WGTRI is required. (See the RAMFIL description.) The number of records to be allocated is as follows:

$$\text{Number of \#WGTRI records} = ((n \times 28 + 4059) / 4060) + 1$$

where n = value specified for NUAT.

### **M3270=768|n**

Maximum 3270 message length, including all control characters. (The maximum is 4KB, although it is not recommended to specify more than 3600.)

**Note:** The following parameters are for EP support only.

### **SUBCH=(sc-ad)**

Up to eight hexadecimal subchannel addresses (01-FF) to define the EP communication controller native subchannel address table. In addition, their associated channel adapter types must be defined by:

- Coding a dash immediately after the 2 hex character subchannel; and
- Coding a 1 for adapter type 1 or a 4 for adapter type 4.



There is a unique subchannel address for each EP communication controller. For example to define subchannel 01F having an adapter type 1 code as follows SUBCH=(1F-1).

**Note:** The number of addresses in the SUBCH parameter governs the number of EP communication controllers on the system. The total number of entries in SUBCH must not be greater than N2703.

**MAXEP=*n***

Maximum number of EP load modules retained on file (1-52). For more information about EP load modules, see the *EP for 3705 Generation and Utilities Guide*.

**MAXDP=0|*n***

Maximum number of EP dumps retained on file (1-99). This defines the number of EP dumps that will be maintained on TPF files before purging to tape.

**DIAG**

Specify one of the following:

- Y6** 16 bit or basic addressing for this CU. Maximum of 48K of storage installed.
- Y8** 18 bit or extended addressing for this CU. More than 48K of storage installed.
- NO** No diagnostic will be included for this system. Up to eight diagnostic specifications to define the diagnostic available for each subchannel may be specified. DIAG parameters must be specified in the same order as the associated subchannel. If the diagnostic parameter is omitted all subchannels will default to NO. If DIAG is coded for selected subchannels the rest will default to Y6. If DIAG=NO is coded all subchannels will default to NO. If Y6 or Y8 is coded for any subchannel NO cannot be coded in the same diagnostic parameter. NO is a system default, therefore, cannot be specified for each channel.

**D3270=NO|YES**

Specifies whether 3270 local lines are defined.

## Examples

None.

## References

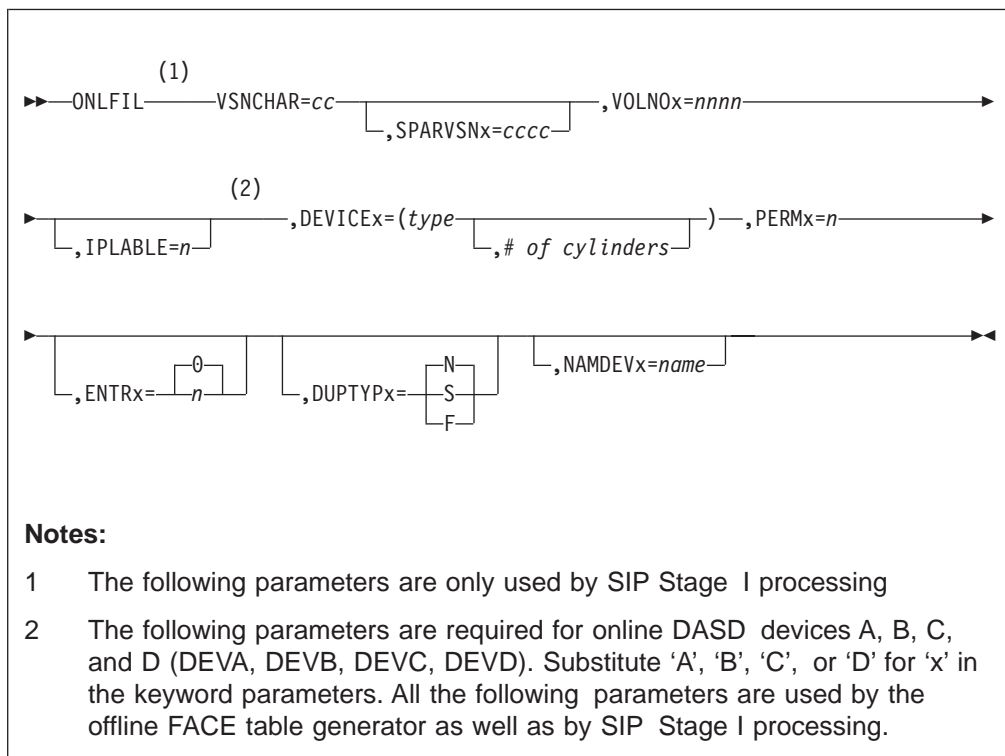
None.

## ONLFIL

The ONLFIL macro is used to specify the characteristics of the DASD devices that will be online and under direct control of the system. ONLFIL macro is required. Before coding this macro it is recommended that the user refer to *TPF Concepts and Structures*.

Up to four DASD devices may be coded using parameters DEVICEA= through DEVIDE=. The parameter DEVICEA= must be coded for a DASD device. Any further DASD devices must use the next alphabetic parameter. For example, if you have two DASD devices then you must code parameters DEVICEA= and DEVICEB=. Combinations such as DEVICEA= and DEVIDEC= or DEVICEB=, DEVIDEC=, and DEVIDE= are not allowed.

## Format



### VSNCHAR=cc

Specifies the first two characters of the six-character volume serial numbers (see the VOLNOx parameter for the description of how the last four characters are coded). These two characters must be alphabetic and they will be used in the VSNs of all online modules, as well as the copy module and the loader general file. While the first two characters of a subsystem's VSN must be identical for all the DASD allocated to it, the first two characters must be unique across subsystems.

### IPLABLE=n

Defines the number of primary and duplicate pairs of DEVA modules that are capable of being IPLed. *n* is an integer between 1 and 255 but not greater than the highest device A primary module.

The following convention is used in describing the remaining DASD parameters: the character identifying the DASD device (A, B, C, or D) is represented in the

parameter names by a lowercase 'x'. For example, DEVICEx becomes DEVICEA for device A, DEVICEB for device B, and so on.

Devices must be coded in "sequential series." In other words, if you are coding parameters for only one device, you must specify this device as device 'A'. If you are coding two devices, you must specify devices 'A' and 'B'. Three devices must be A, B, and C, and four devices must be A, B, C, and D. (For example, if you have three devices, you cannot code them as devices A, C, and D.)

You can code the following parameters in any order, as long as you code all the required parameters (those without defaults) for each device. The parameters with defaults are ENTRx, DUPTYPx, and NAMDEVx.

**DEVICEx=type[,# of cylinders]**

Specifies one of the supported DASD device types as follows:

Device Type	Default # Cylinders
3350 - 3350 Model 2 or Model 2F	555
3375 - 3375 Model 1 or Model 1F	959
3380 - 3380 all models	885
3390 - 3390 all models	1113
9345 - 9345	1440

The number of cylinders indicates the maximum number of cylinders to be used by the specified device type. See Table 7 on page 79 for more information.

**Notes:**

1. 3390 devices are set at installation time to either native 3390 mode or 3380 emulation mode. A 3390 running in 3380 emulation mode must be coded as DEVICEx=3380 for both types of emulation. TPF supports 2 types of 3380 emulation.
  - a. Track compatibility emulation, which is supported on 3990 control units that are not using CFLF. For sample JCL to set or change the mode of a 3390, see the description of the ICKDSF INSTALL command (SETMODE option) in the *DSF User's Guide and Reference*.
  - b. Record format emulation, which is used with CFLF. 3380 emulation in this case is set by using Licensed Internal Code (LIC) that runs directly on the 3990.
2. The volume table of contents (VTOC) dataset will be placed at one of the following locations:
  - The last track of the cylinder that was indicated.
  - The location that was specified by coding the RAMFIL macro statement with the VTOC parameter specified.
  - Cylinder 4369, head 0. This is the default location if the RAMFIL macro statement with the VTOC parameter specified is not coded and the number of cylinders is more than 4369.

**PERMx=n**

Specifies the number of device x online modules for fixed and pool records on both prime and dup mods.

**Note:** The total number of online plus general file devices cannot exceed 3999 (or 999 if the system is in migration mode).

**VOLNOx=nnnn**

Specifies the numeric portion of the six-character starting volume serial number

## ONLFIL

for DEVICEx. The parameter accepts a 1-to-4 digit decimal number (or a 1-to-3 digit decimal number if OLDVSN was coded).

### Notes:

1. The numeric characters cannot be all zeros, as VSN zero has been reserved for the copy module. Also the numeric portions of VSN range should not overlap the range of another device type, or the loader general file. SIP will assign sequential numbers starting with the specified number for the device type.
2. The VSNs for the rest of a device type's DASD are calculated by adding one to the starting number for each DASD allocated. The last VSN created will be:

$$\text{VOLNOx} + \text{PERMx} + \text{EXTRx} - 1$$

### EXTRx=0|n

Specifies the number of extra device x file status table slots to be reserved for future expansion.

### DUPTYPx

Specifies the duplication status for device x online modules.

- N**      Nonduplicated.  
**S**      Selectively duplicated.  
**F**      Fully duplicated.

### NAMDEVx=name

This parameter is used to assign a symbolic four-character name to a DASD device type. This name is used as a device type identification in the DASD pool program messages. Each DASD device type must have a unique identification.

For example, if the user's system is to contain 3380 type DASDs as both device 'A' and device 'B' (e.g., pools formatted differently) then this parameter would have to assign unique names for device 'A' and 'B', such as NAMDEVA=DEVA and NAMDEVB=DEVB.

*The default is whatever was coded for the DEVICEx parameter.*

### SPARVSNx=cccc

Specifies the first four alphameric characters of the volume serial numbers assigned to spare volumes to be used for copying from the online modules in the event of a DASD failure. VSNs starting with this character string will be ignored during disk roll call.

For example, specifying this parameter as 'SPAR' will allow you to attach any number of devices with the VSN 'SPAR00' without disrupting the IPL with unwanted duplicate VSN error messages. Since each logical device type of each MDBF subsystem may be formatted with its own unique record allocation, a different spare VSN may be coded for each device type of each MDBF subsystem. This will allow operations to more easily identify the appropriate spare to use for a given device, when multiple disk formats are used.

## Examples

The following example specifies four DASD devices. Two 3390s are to be assigned as device A permanent modules and two 3380s are to be assigned as device B permanent modules. Both device A and B are to have four extra file status table slots reserved. Both device A and B are to be selectively duped. Device A has a

starting VSN of SP0003 and device B has a starting VSN of SP2010. Device A is to be identified as DEVA and device B is identified as 3380 by the DASD pool program messages.

```
ONLFIL VSNCHAR=SP, X
      DEVICEA=(3390,1440),PERMA=2,EXTRA=4, X
      VOLNOA=3,DUPTYPA=S, X
      NAMDEVA=DEVA X
      SPARVSNA=SPRA, X
      DEVICEB=(3380,885),PERMB=2,EXTRB=4, X
      VOLNOB=2010,DUPTYPB=S, X
      SPARVSNB=SPRB
```

References

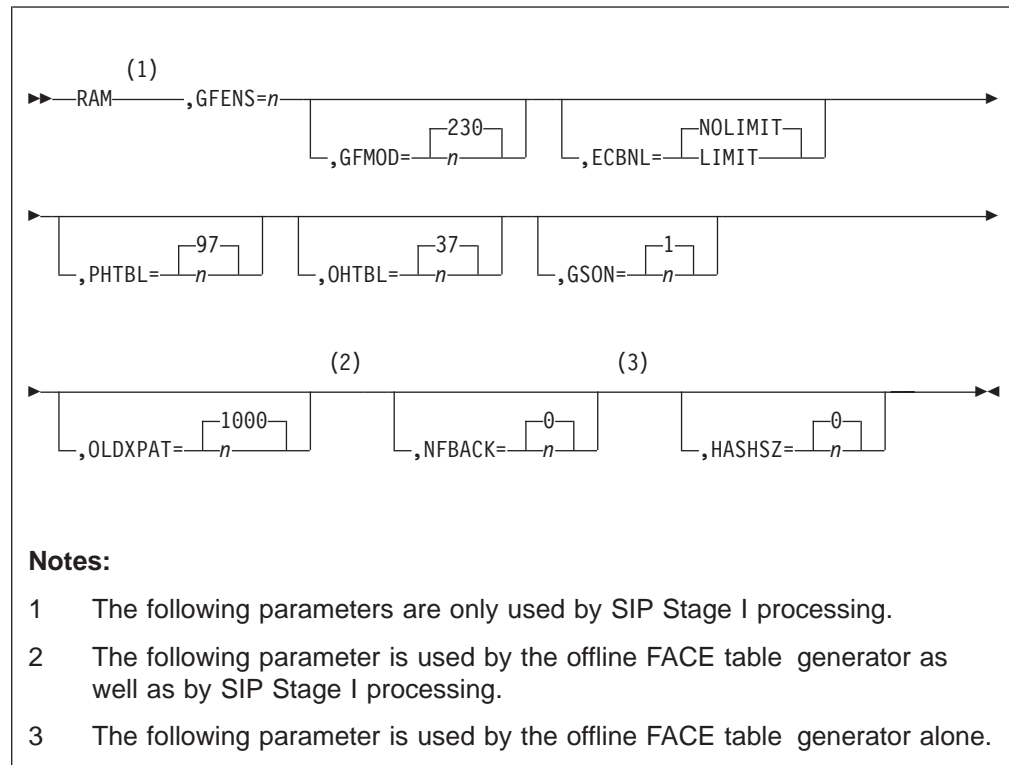
None.

## RAM

The RAM macro is used to specify the resource allocation characteristics.

The RAM macro is required.

## Format



### **GFENS=*n***

Specifies the number of slots to be created in the General File Module Table (must be greater than the highest data set number specified in the GENFIL macro but less than 61). This value also represents the number of general file pseudo module numbers allowed on the system, and it is used to determine the number of Module File Status Table entries to generate for use of general files.

### **GFMOD=230|*n***

Specifies the starting general file symbolic module number (must be three decimal characters).

### **ECBNL**

Designates whether the number of program nesting levels is limited or not through the use of overflow blocks.

#### **NOLIMIT**

The number of program nesting levels is unlimited through the use of overflow blocks.

**LIMIT** The number of program nesting levels is limited to the area provided in the ECB.

### **PHTBL=97|*n***

Number of primary hold table entries. To take full advantage of the system's hashing algorithm, this should be a prime number.

**Note:** In a system with the multiple database function (MDBF) this parameter is necessary only for the basic subsystem generation. (BSSGEN=YES in the SSDEF macro.) All subsystems will use the value from the basic subsystem.

#### **OHTBL=37|n**

Number of overflow hold table entries. To take full advantage of the system's hashing algorithm, this should be a prime number.

**Note:** In a system with the multiple database function (MDBF) this parameter is necessary only for the basic subsystem generation. (BSSGEN=YES in the SSDEF macro.) All subsystems will use the value from the basic subsystem.

#### **GSON=1|n**

Specifies the number of Module File Status Table entries to be created for use with general datasets. 'n' must be in the range 0-3997.

**Note:** The summation of GFENS and GSON may not exceed 3998. Also, keep in mind that the ZAVSN/ZDVSN commands require an available general dataset entry in the Module File Status Table (MFST).

#### **OLDXPAT=1000|n**

Specifies the number of extra program allocation table (PAT) slots to allocate for the E-type loader. Use the following formula to help determine how many extra PAT slots are needed:

$$A(B + C(D - 1))$$

Where:

- A** Maximum number of active loadsets for a given processor.
- B** Average number of programs in a loadset.
- C** Average number of I-stream unique programs in a loadset.
- D** Number of I-streams in the system.

In addition to using the formula, it is important to remember that when a loadset is deactivated, the PAT slots used by that loadset remain in use until there is no more activity on the loadset. Therefore, consider increasing the number of PAT slots determined by the formula by approximately 10% to accommodate this situation.

This parameter is subsystem unique. The value of this parameter is stored in keypoint A (CTKA).

#### **NFBACK=0|n**

Specify the number of fallback keypoint areas desired in the #KEYPT record area. This option is for use in an MDBF system where the basic subsystem is generated on a small number of packs. If specified in a non-MDBF system, the fallback areas cannot be used to IPL the system. This parameter is supported for coexistence of systems with and without 32-way loosely coupled processor support. Once migration to 32-way loosely coupled processor support is completed, the number of fallback extents will be controlled by the #KFBXn RAMFIL definitions in the FACE table (FCTB). See *TPF Main Supervisor Reference* for more information about keypoints.

#### **HASHSZ=0|n**

Defines the number of 4-byte entries of the hash table used for the symbolic

## RAM

resolution of FACE IDs by program FACS. If HASHSZ is not specified or is coded as 0, the offline FACE table generator will attempt to build a table with no synonyms. The table size is unpredictable, although it is less than 32 767 entries. Coding a valid hash table size will fix the size of the hash table. The smallest value that may be coded for this parameter is one more than the number of unique record types. If the value is coded, it should be coded as a prime number. The largest value for this parameter is 128KB-1 (131 071).

## Examples

None.

## References

None.



---

## RAMEND

This macro must be coded after the last RAMFIL macro. It contains no parameters and is used by the assembler to signal the end of all the RAMFIL macros that are coded.

---

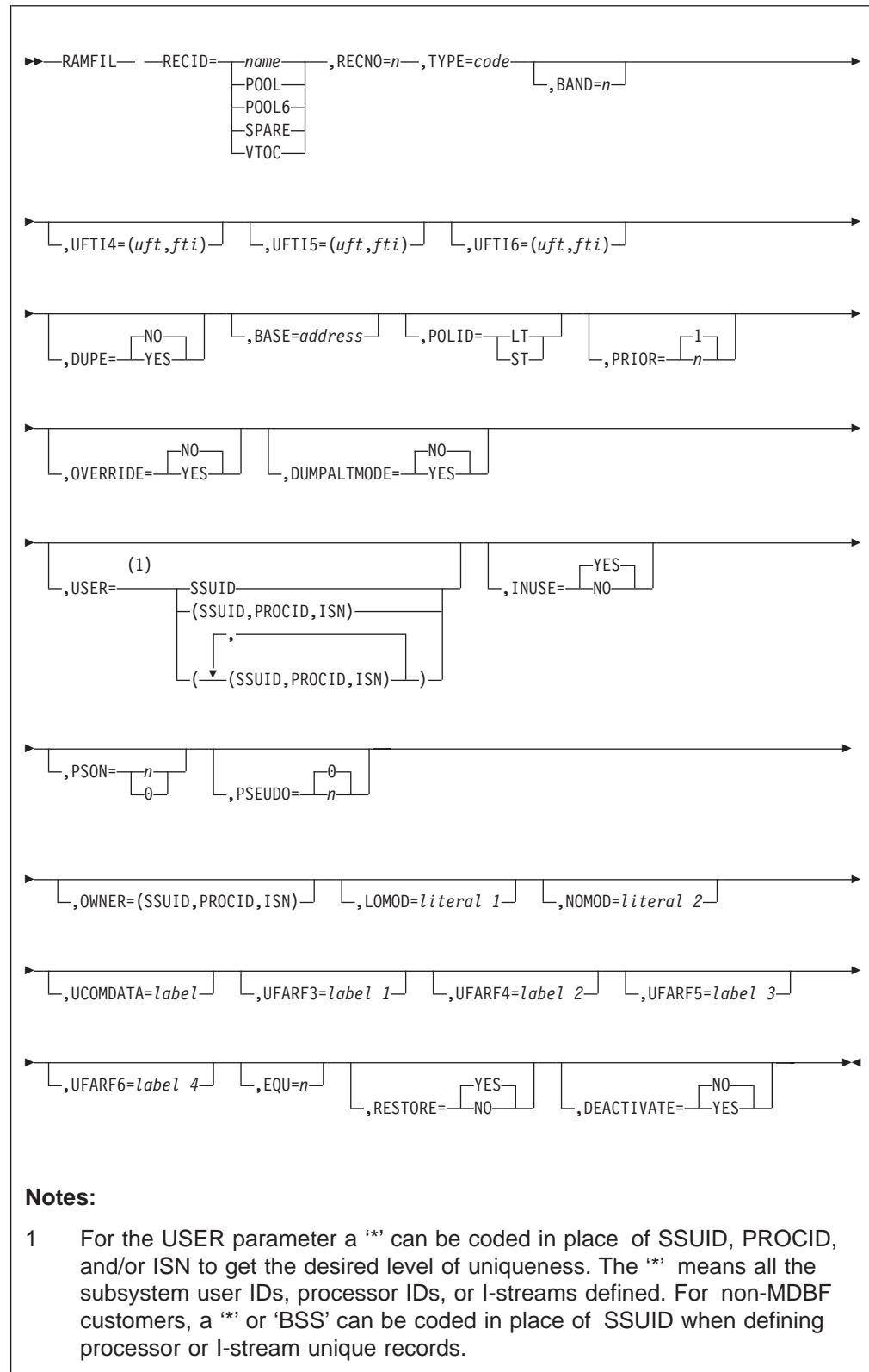
## RAMFIL

The RAMFIL statement is used to specify characteristics of fixed, pool and spare records for TPF database storage devices. The FACE Table is created as a direct result of coding this statement. The order of record groups on disk is determined by the sequence of RAMFIL statements coded in SIP stage I. The RAMFIL statement is required and must be coded in ascending sequence of your file layout. It is recommended that you review information about data organization in *TPF Concepts and Structures* before coding the RAMFIL statement.

Certain fixed file records are required to have RAMFIL statements coded for them. The specific ones depend on the particular support functions to be included in the system. See Table 16 on page 105 for the required record types.

See “Pool File Storage” on page 68 for information about determining the minimum number of pool directories.

## Format



### RECID

Specify one of the following:

## RAMFIL

*name* Identifies the record type name used to retrieve the record. For example:  
RECID=#WAARI.

**Note:** Record types that do not start with # will not work with certain TPF messages.

**POOL** Indicates that this is a file pool.

**POOL6** Indicates that this is a FARF6 file pool.

**SPARE** This value reserves space on a volume.

**VTOC** This value reserves space on a volume for the volume table of contents (VTOC) data set. The size of the VTOC data set is 1 track. The BASE parameter must be coded when the VTOC parameter is coded. The last valid track location is cylinder 4369, head 0.

### RECNO=*n*

Number (decimal) of records to be allocated for this record type. Must be greater than or equal to 1. If the record type is a vertically allocated record, the RECNO reflects the records reserved on a single module only. See Table 18 on page 111 for a list of vertical record types.

### TYPE=*code*

Specifies the size and storage device type for records.

**Note:** RAMFIL statements for a specific device type must be grouped together.

**SSx** Small record residing on DASD Device x (DEVx). x = TPF devices A, B, C or D

**LSx** Large record residing on Device x (DEVx). x = TPF devices A, B, C or D

**4Sx** 4K records residing on Device x (DEVx). x = TPF devices A, B, C or D

**Note:** Refer to ONLFIL macro for DASD devices (Device A, Device B, Device C, Device D).

### BAND=*n*

BAND numbers are only used by FARF3.

The parameter specifies one or more 'band' numbers to be assigned to the RECID specified by this RAMFIL statement. Each band number must be specified as a decimal number from 0 to 4095, inclusive. This operand must be coded for all RAMFIL macros with PRIOR=1 that are to have FARF3 addresses generated, except those with RECID=SPARE or POOL. This parameter is not valid when RECID=SPARE or RECID=POOL. This parameter is ignored if STAGE=FARF45 is coded on the UFTFTI statement.

Each band number value specified represents 65 536 records. The band number is used in conjunction with the low order 16 bits of a record number to form a unique fixed file address for a record by the FACE program. By keeping the band numbers for a record type constant from one system generation to the next, the records can be physically relocated without the fixed file FARF3 address changing. Thus embedded file addresses need not be changed.

If a record type requires more than one (1) band number (for example, has more than 65 536 records including lower priorities or expansion beyond 65 536 records is anticipated) the band numbers may be divided and coded as:

A range	BAND = (100–102)
A sublist	BAND = (100,101,102,77)
A combination	BAND = (100–102,77)

The band numbers are used in the order in which they are specified. That is, the first band number is used for record numbers 0–65535, the second band number for record numbers 65536–131071, and so on.

A band number can only be used once for each subsystem user. However, band numbers for each subsystem user are independent of any other subsystem user. Band numbers may be shared by subsystem user unique records only. Band numbers may not be shared by processor and I-stream unique records. Therefore, when defining processor and I-stream uniqueness, it is necessary to select a new band number for each instance of uniqueness.

#### **UFTI4, UFTI5=(*uft*,*fti*)**

Specifies which *uft/fti* combination to use for allocating addresses. Each *uft/fti* combination is specified, in decimal, as an ordered pair. The range for *uft* is 0 to 63. The range for *fti* is dependent on the *fti* length for the given *uft*. Refer to the UFTFTI macro “UFTFTI” on page 318 and UFT/FTI examples “UFT/FTI Examples” on page 93 for more information.

The *uft/fti* combination is used in conjunction with a record number to form a unique FARF address for a record. By keeping the *uft/fti* combinations constant from one system generation to the next, records can be physically relocated without the FARF addresses changing. Thus, embedded file address need not be changed.

The *uft/fti* combination (0,0) is illegal, even though the combinations (0,x) and (x,0), where x > 0, are valid.

If a record type requires more than one (1) *uft/fti* combination, a list of *uft/fti* combinations can be coded. For example:

UFTI4=( (3,12) , (5,24) , (3,10) )

*uft/fti* combinations are used in the order in which they are specified, and can be used only once. For instance, notice that the *uft/fti* combinations are unique in the UFTI4 example statement. The *uft/fti* combinations must also be unique between the UFTI4 and UFTI5 parameters. They must also be unique between RAMFIL statements.

We recommend that the lowest FTIs be used first for each UFT to conserve space in the FACE table.

This parameter may be coded only for PRIOR=1 fixed file records and for the lowest pool ordinal number (PSON) for a given pool type. The UFTI4 parameter is required if the user wants to define a FARF4 address for a given RECID. The UFTI5 parameter is required if the user wants to define a FARF5 address for the RECID.

The user may want to code UFTI5 in a STAGE=FARF34 FACE table (refer to the UFTFTI statement “UFTFTI” on page 318). Although no FARF5 addresses are created, the ability to migrate will be verified by the offline FACE table generation. This means that the user can put UFTI5 parameters their RAMFIL statements and have the offline table generator check it for them, even though no FARF5 addresses are actually generated. By verifying their UFTI5

## RAMFIL

statements the user can ensure the FARF5 migration path is intact while still laying out the database for FARF34 migration.

The assembler only allows 255 characters to be coded for a sublist parameter. If more than 255 characters, including parentheses and commas, are coded for UFTI4 or UFTI5, the statement will not assemble. Because information from RAMFIL is not needed during SIP stage I assembly, statements that do not assemble can be commented out. You will need to uncomment these statements before the FACE Table Generator reads them.

### **UFTI6=(*uft*,*fti*)**

Specifies which *uft/fti* combination to use for allocating FARF6 addresses. The UFT/FTI combinations coded for UFTI6 are totally independent from the UFT/FTI combinations coded for UFTI4 and UFTI5. Each *uft/fti* combination is specified, in decimal, as an ordered pair. The range for *uft* is 1 to 65535. The range for *fti* depends on the *fti* length for the given *uft*. See "UFTFTI" on page 318 and "UFT/FTI Examples" on page 93 for more information.

The *uft/fti* combination is used with a record number to form a unique FARF address for a record. By keeping the *uft/fti* combinations constant from one system generation to the next, records can be physically relocated without the FARF addresses changing. Because of this, embedded file addresses do not need to be changed.

All *uft/fti* combinations of the form 0,x are illegal.

If a record type requires more than one (1) *uft/fti* combination, a list of *uft/fti* combinations can be coded. For example:

UFTI6=((3,12),(5,24),(3,10))

*uft/fti* combinations are used in the order in which they are specified and can be used only once. For example, notice that the *uft/fti* combinations are unique in the UFTI6 example statement. They must also be unique between RAMFIL statements.

We recommend that the lowest FTIs be used first for each UFT to conserve space in the FACE table.

This parameter is valid only for the lowest PSON for a 4-K duplicated long-term pool type. The UFTI6 parameter is required if you want to define a FARF6 address. If the UFTI6 parameter is coded, the UFTI4 and UFTI5 parameters cannot be coded.

### **DUPE**

Specify one of the following:

**NO** Indicates nonduplicated record-type.

**Note:** In a nonduplicated system DUPE=NO should be coded for all record types. In a fully duplicated system DUPE=YES should be coded. Short-term pools must always be specified as DUPE=NO. See "Record Duplication" on page 98 for more information.

**YES** Indicates that this is a duplicated record.

### **BASE=address**

Specifies the record address (in decimal). The format is the cylinder and head number (cccchh).

**Note:** The BASE parameter is required for:

1. POOL and FARF6 pool records
2. Whenever the TYPE or DUPE parameter is different from the TYPE or DUPE parameters on the previous RAMFIL
3. For all vertical records (#CTKX, #CIMRx, #IPLx, #KEYPT).

### **POLID**

Specify one of the following:

**LT** Long-term file storage pool.

**ST** Short-term file storage pool.

If POLID is set to ST then the DUPE parameter must be NO.

### **PRIOR=1|n**

Priority number for fixed file records spread across several RAMFIL statements (those that will be chained to others according to PRIOR sequence). Each record type uniqueness group must have a PRIOR=1 (specified either explicitly or by default). The default is 1 and represents a prime entry.

The higher number indicates a lower priority.

The PRIOR parameter does not apply to POOL and SPARE records.

Duplicate RECID names must not be specified with duplicate priorities within the same uniqueness group. No gaps may exist in the PRIORs defined for a particular RECID uniqueness group.

#### **Example:**

```
RAMFIL RECID=#FRED,RECNO=20,TYPE=LSA,PRIOR=3.
RAMFIL RECID=#FRED,RECNO=10,TYPE=LSA,PRIOR=2.
RAMFIL RECID=#FRED,RECNO=30,TYPE=LSA,PRIOR=1,BAND=100
RAMFIL RECID=#FRED,RECNO=15,TYPE=LSA,PRIOR=4.
```

### **VERRIDE=NO|YES**

PSO in order coding. If YES is specified, a given RAMFIL statement may be coded with a PSO out of order. If NO is specified, each PSO must be greater than the PSO of the previous pool of the same type. The OVERRIDE parameter is used with POOLS only.

**Note:** The OVERRIDE parameter is not valid for FARF6 pools. PSOs can be coded in any order for FARF6 pools without coding the OVERRIDE parameter.

### **DUMPALTMODE**

Exposes improperly or unmigrated FARF addresses.

**NO** The system will not dump when processing FARF addresses outside the currently selected dispensing mode.

**YES** The system will dump when processing a FARF address for this RECID uniqueness group not in the selected FARF dispensing mode.

DUMPALTMODE is only valid on the lowest PSO of each pool type and is not valid on RAMFILs with PRIORs greater than 1.

**Note:** DUMPALTMODE is not valid for FARF6 pools.

### **USER=attribute**

An attribute can have one of the following forms:

- SSUID
- (SSUID,PROCID,ISN)

## RAMFIL

- ((SSUID,PROCID,ISN)[,(SSUID,PROCID,ISN)])

An attribute defines the Subsystem Users, Processors, and I-streams associated with a fixed record ID.

An asterisk '\*' is used to specify all Subsystem Users, Processors, or I-streams not yet specified in prior RAMFIL statements. The '\*' may take the place of either or all of the attribute specified previously. Unspecified attributes default to '\*'. In other words the default is USER=(\*,\*,\*).

The SSUID must have been specified in the SSUID parameter of SSDEF.

For non-MDBF customers a '\*' or 'BSS' can be used when specifying SSUID.

The PROCID must have been specified in the SYSID parameter of CONFIG.

The ISN must take a value between 1 and 8.

### Notes:

1. An SSUID is required, while the PROCID and ISN specifications are optional. If an I-stream (ISN) is not specified, a processor ID (PROCID) should not be provided either. In this case (where SSUID would appear alone within parentheses), the specification is simply coded as USER=ssu, a subsystem user ID as found in the SSDEF macro.
2. The USER parameter should not be coded for POOL records (RECID=POOL).
3. The assembler only allows 255 characters to be coded for a sublist parameter. If more than 255 characters, including parentheses and commas, are coded for USER, the statement will not assemble. Because information from RAMFIL is not needed during SIP stage I assembly, statements that do not assemble can be commented out. You will need to uncomment these statements before the FACE Table Generator reads them.

The following are examples of the USER parameter.

1. USER=WP2 and USER=(WP2,\*,\*) are functionally equivalent. The record type is unique to SSU WP2 on all processors and I-streams.
2. USER=(WP1,\*,2) the record type is unique to SSU WP1, on all processors, in I-stream 2.
3. USER=((WP1,\*,\*), (WP2,\*,\*), (HPN,B,\*), (WP3,D,6)) the record type is unique to SSU WP1 on all processors and I-streams, to SSU WP2 on all processors and I-streams, to SSU HPN on processor 'B' on all I-streams, to SSU WP3 on processor 'D' on I-stream 6.

### INUSE=YES|NO

Specifies that a record is in use.

This parameter is only valid on PRIOR=1 fixed file RAMFIL statements.

### PSON=0|n

Specifies the first pool database ordinal number to be used for the pool segment being defined.

**Note:** This parameter is valid only when RECID=POOL is specified for a DASD device.



The PSON value specified must not overlap any PSON range for the same DASD pool type. This is true even when going from one device type to another. If PSON is not specified, then SIP will compute PSON as follows:

PSON = highest PSON already defined  
for this pool type plus 1

**Note:** The PSON must be specified for FARF6 pools.

For STAGE34 database definitions, PSON recognizes only two categories of pool sizes: small and large. For the purposes of calculating the PSON, 381-byte records are considered “small” while both 1055- and 4K-byte records are considered “large.” Therefore, for FARF3 database definitions, 1055 and 4K pool segment PSONs may not overlap with each other.

PSONs must be coded in ascending order for FARF3, FARF4, or FARF5 pool types in a stage I deck unless the OVERRIDE parameter is coded. For FARF6 pools, PSONs can be coded in any order without specifying the OVERRIDE parameter.

Although the SIP calculates a default, it is recommended that this parameter be coded. In Stage FARF3/4 large and 4KB pool records share PSONs, but in Stage FARF4/5 they do not share PSONs. This could lead to the PSONs changing for large and 4KB pool records when the stage is changed from FARF3/4 to FARF4/5.

#### **PSEUDO=0|n**

Specifies the number of pseudo modules to be used in the allocation of the pool record addresses for the pool segment specified by this RAMFIL statement. Pseudo modules are modules that are not currently present on the online system but may be added in the future for expansion of the DASD pools without having to do a file reorganization. Pseudo records are records contained on pseudo modules. Like pseudo modules, pseudo records are for expansion purposes only. The number of pseudo records is calculated by multiplying the number of pseudo modules for a given record type by the number of records of that type (RECNO).

The value specified for this parameter must be a multiple of 2 (or 0).

**Note:** Another method of expanding the online files which does not require the expansion modules to be pre-planned at system generation time is the use of the Data Base Reorganization Program. This program provides for reorganization of both the fixed and pool records. For further information about database reorganization, see the *TPF Database Reference*.

If a value greater than 0 is specified, the DASD pool directories will be generated to include pool records residing on the pseudo modules. However, the pseudo module pool record addresses will be in a DEACTIVATED state and, therefore, will not be dispensed by the online system until the file pool programs are told to activate the pseudo modules.

**Note:** See *TPF Operations* for information about how to activate pools allocated on the pseudo modules.

The number of pool records for this segment that can be contained on the pseudo modules must be added to the RECNO parameter value to determine the pool ordinal number range for this segment. See the PSON parameter

## RAMFIL

description above. The pool type (duped/nonduped) and device type duplication (nonduped, fully duped or selectively duped) must be considered in determining the number of pseudo pool records.

**Note:** If this parameter is specified, the EXTRx parameter of the ONLFIL macro should also be coded to facilitate the physical addition of the pseudo modules to the online system in the future.

**OWNER=***triplet***ssu**

Where a triplet is of the form:

(SSUID, PROCID, ISN)

The triplet defines the owning subsystem user, processor, and I-stream combination. The owner of a record must also be specified as a user of the record. An SSUID is required, while the PROCID and ISN specifications are optional. When SSUID would appear alone in parentheses, the specification is simply coded as OWNER=ssu, a subsystem user ID as found in the SSDEF macro. For non-MDBF customers, a \* or BSS can be used for SSUID. Information specified by this parameter is used by Data Base Reorganization and Recoup.

The default is the first triplet specified or implied by the USER parameter.

This parameter is only valid for fixed file records. The owner of a set of pools is the first subsystem user, processor, and I-stream.

**LOMOD=***literal 1*

Literal 1 is the starting relative module number for the pool segment.

**The default value is 0.**

This parameter is only valid for POOL records (RECID=POOL, RECID=POOL6). The value coded for this parameter cannot exceed the number of prime and pseudo modules in the system.

**NOMOD=***literal 2*

Literal 2 is the number of pool modules to spread this pool segment over starting at the module specified by LOMOD.

The default is the number of prime + pseudo modules.

This parameter is only for POOL records (RECID=POOL). The sum of LOMOD and NOMOD must not exceed the number of prime and pseudo modules.

**UCOMDATA=***label*

The UCOMDATA parameter attaches user data to the Split Chain Header associated with a RECID. This parameter is valid only on RAMFILs with PRIOR=1 or the lowest PSON for a given POOL type. The label must be a valid entry point in the user data, because it becomes a VCON in the FACE table. The VCON is resolved by the MVS linkage editor. The value coded for this parameter must be the same for all PRIOR=1 fixed file records of the same record id (RECID).

The effect of this mechanism is to provide an extension of information associated with a record type. When the record type is accessed, a pointer to the external user data is returned giving a calling program access to that data.

The symbols "CTSD" and "CONK" are reserved and should not be redefined in the external symbol dictionary. This applies to the UFARFx parameters as well.

This parameter is not valid for SPARE records.

**UFARF3=*label* 1**

The UFARF3 parameter attaches user data to the FARF3 split(s) generated by the RAMFIL for which it is coded. The label must be a valid entry point in the user data, because it becomes a VCON in the FACE table. The VCON is resolved by the MVS linkage editor.

This parameter is not valid for SPARE records.

**UFARF4=*label* 2**

The UFARF4 parameter attaches user data to the FARF4 split(s) generated by the RAMFIL for which it is coded. The label must be a valid entry point in the user data, because it becomes a VCON in the FACE table. The VCON is resolved by the MVS linkage editor.

This parameter is not valid for SPARE records.

**UFARF5=*label* 3**

The UFARF5 parameter attaches user data to the FARF5 split(s) generated by the RAMFIL for which it is coded. The label must be a valid entry point in the user data, because it becomes a VCON in the FACE table. The VCON is resolved by the MVS linkage editor.

The effect of the UFARF*x* parameters is to make user specified information available on an addressing mode basis.

This parameter is not valid for SPARE records.

**UFARF6=*label* 4**

The UFARF6 parameter attaches user data to the FARF6 splits generated by the RAMFIL statement for which it is coded. The label must be a valid entry point in the user data because it becomes a VCON in the FACE table. The VCON is resolved by the MVS linkage editor.

The effect of the UFARF*x* parameters is to make user-specified information available on an addressing mode basis.

This parameter is not valid for SPARE records and is only valid when the UFTI6 parameter is also coded.

**EQU=*n***

The EQU parameter sets the SYSEQC equate value for the RECID to *n*. This parameter is valid for fixed and pool records. *n* must be in the range 0...12287. The purpose of this parameter is for compatibility with older databases requiring segments to be assembled against specific SYSEQC values for record IDs. If not specified, a default is chosen.

**RESTORE=YES|NO**

This parameter is used to indicate whether or not the record type should be restored. The default is YES except for the following record types whose default is NO: #DBRRI, #RSTRI, #KEYPT, #TPLBL, #CIMR1–8, #CTKX, #KBA, #KSA1–8, #PROG1–8, #OLD1–8, #PVR1–8, #XPRG1–8, #RLOG1–8, and #IPL1–4. This parameter is not valid for POOL or SPARE record types and may only be coded on PRIOR=1 RAMFILs. If it is coded on PRIOR=1 RAMFILs, RESTORE must be coded the same for each PRIOR=1 RAMFIL for a particular record type.

**DEACTIVATE=NO|YES**

This parameter is used to indicate if a pool segment is going to be deactivated. If YES is specified and a pool deactivation is run, these pool addresses will no longer be dispensed by GETFC macro calls but will continue to be decoded. This parameter is only valid for pool addresses.

## RAMFIL

### RAMFIL Parameter Rules

Table 23. RAMFIL Parameter Rules

Parameter	Fixed	Pool	Spare or VTOC
RECID	Required	Required	Required
RECNO	Required	Required	Required
TYPE	Required	Required	Required
DUPE	Allowed See note 2.	Allowed See note 2.	Allowed See note 2.
BASE	Allowed See note 1.	Required	Allowed See note 1.
PRIOR	Required See note 2.	Invalid	Invalid
PSON	Invalid	Allowed	Invalid
USER	Allowed See note 2.	Invalid	Invalid
INUSE	PRIOR=1 See note 2.	Invalid	Invalid
BAND	PRIOR=1 See note 3.	Invalid	Invalid
UFTI4	PRIOR=1 See note 3.	Low PSON See note 3.	Invalid
UFTI5	PRIOR=1 See note 3.	Low PSON See note 3.	Invalid
UFTI6	Invalid	Low PSON See note 3.	Invalid
POLID	Invalid	Allowed See note 2.	Invalid
PSEUDO	Invalid	Allowed	Invalid
OVERRIDE	Invalid	Allowed	Invalid
DUMPALTMODE	PRIOR=1 See note 2.	Low PSON See note 2.	Invalid
OWNER	PRIOR=1 See note 2.	Invalid	Invalid
LOMOD	Invalid	Allowed	Invalid
NOMOD	Invalid	Allowed	Invalid
UCOMDATA	PRIOR=1 See note 4.	Low PSON	Invalid
UFARF3	Allowed	Allowed	Invalid
UFARF4	Allowed	Allowed	Invalid
UFARF5	Allowed	Allowed	Invalid
UFARF6	Invalid	Allowed	Invalid
EQU	Allowed See note 5.	Allowed See note 5.	Invalid
RESTORE	PRIOR=1 See note 4.	Invalid	Invalid
DEACTIVATE	Invalid	Allowed	Invalid

Table 23. RAMFIL Parameter Rules (continued)

Parameter	Fixed	Pool	Spare or VTOC
<b>Notes:</b> <ol style="list-style-type: none"> <li>1. A BASE is always allowed but it is required if the RAMFIL it is coded on has a different type, size or duplication from the previous RAMFIL.</li> <li>2. Optional unless the default is unacceptable in which case it must be coded.</li> <li>3. The parameter is required if you want splits of a the particular address type generated for this RECID.</li> <li>4. If the parameter is coded, the value coded on it must match all the same parameter on all the other PRIOR=1 RAMFILs for this RECID.</li> <li>5. If coded, the parameter must appear on the first RAMFIL in the stage I deck for the particular RECID or pool type.</li> </ol>			
<b>Required</b>	The parameter must be coded.		
<b>Allowed</b>	The parameter may be coded.		
<b>Invalid</b>	The parameter must not be coded.		
<b>PRIOR=1</b>	The parameter may only be coded on a RAMFIL with PRIOR=1 coded.		
<b>Low PSN</b>	The parameter may only be coded on the lowest PSN for this particular pool type.		

## Examples

None.

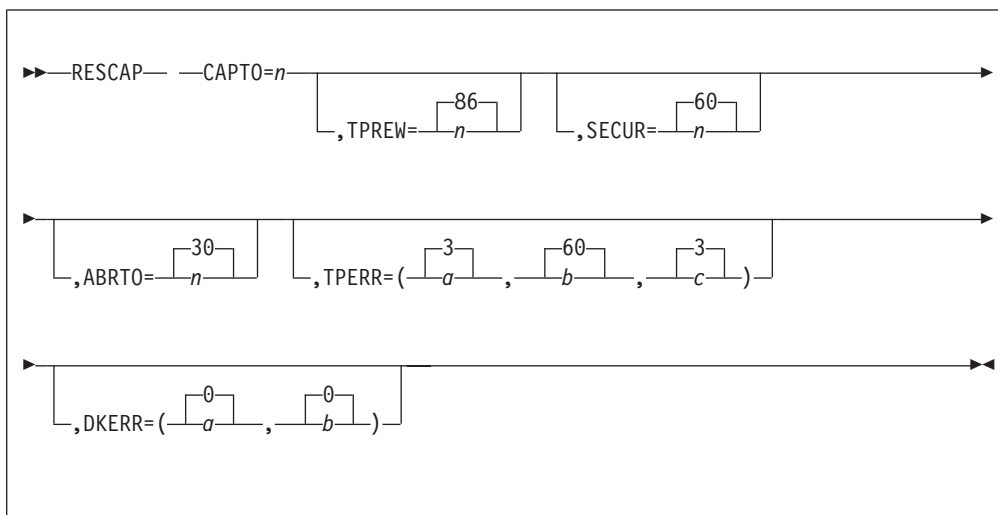
## References

- *TPF Database Reference*
- *TPF Concepts and Structures*.

## RESCAP

The RESCAP macro is used to specify error counts, channel utilization percents and other variable data pertaining to the file capture/restore facility. It is also used to input the channel, control unit and relative speed data for available tape units. RESCAP is required if the capture and restore utility is desired.

## Format



### CAPTO=n

System pause time-out factor in seconds. This value is the maximum amount of time allowed for a capture or restore to complete when a pause message is entered. The value inserted should be greater than the slowest time required to capture/restore a module. For test purposes it has been set at 455 seconds.

### TPREW=86|n

Maximum rewind and unload time in seconds for any tape device which can be utilized by capture/restore. This value should be at least equal to the maximum rewind and unload time for the slowest tape device on the system, plus an operator considerations factor (i.e., time required to mount a new tape). Zero is not valid. **Default is 86 seconds**, the value for an ECCST cartridge.

### SECUR=60|n

Time interval in seconds at which the capture/restore Security program is activated to determine if capture or restore read/write entries are processing longer than expected. Zero is not valid.

### ABRTO=30|n

Maximum amount of time in seconds allowed for entries to abort after an abort message is entered. Zero is not valid.

### TPERR

Tape error counts.

**3|a** Number of times an internal tape mount will be attempted prior to aborting the module. Zero is invalid.

**60|b** Time interval in seconds which must elapse between unsuccessful tape mount retries. This value is used in conjunction with "a" above in attempting to mount tapes. Zero is invalid.

3|c      Number of attempts made by the Restore package to initially read the header of a tape to be restored.

**DKERR**

Disk error counts.

0|a      Number of times capture will retry an individual disk error (in addition to system error retry procedures).

0|b      Number of times restore will retry an individual disk error (in addition to system error retry procedures).

**Examples**

None.

**References**

None.

## RIATA FINISH

### RIATA FINISH

The RIATA FINISH macro call is the third in a series of RIATA calls used to create entries in the record ID attribute table (RIAT).

The RIAT is created by coding a deck of RIATA calls and assembling it. The RIAT deck consists of:

1. A RIATA START call (see “RIATA START” on page 311)
2. A series of RIATA ID= calls (see “RIATA ID” on page 305)
3. A RIATA FINISH call.

### Format



### Entry Requirements

The macro must be coded in the sequence described previously, the RIATA START call, followed by RIATA ID calls, followed by a RIATA FINISH call.

### Return Conditions

Output consists of either the RIAT table or MNOTEs (see *Messages (Online)*). If any of the parameters are coded incorrectly, the calls occur in the wrong order or duplicate IDs are coded. Careful understanding of the RIAT and RIATA input is required to properly interpret RIATA output.

### Programming Considerations

The RIATA macro calls are coded in the RIAT segment, not in the stage I deck.

### Examples

```
RIATA START
RIATA ID=B'00000000100000001',RTP0=(S,ST,A),RTP1=(S,LD,A),    *
      VFAP=IMMED
RIATA ID=X'102',UEXIT=YES,XCP=NO,LOG=YES
RIATA ID=5008
RIATA ID=C'AA',RTP5=(4,LD,A),RTP8=(4,LT,A),VFAP=DELAY
RIATA ID=X'C1C3',VFAF=IMMED,VFAP=IMMED
RIATA ID=C'TC',VFAF=IMMED,VFAP=IMMED,RCSF=CFWD,RCSP=DFW
RIATA FINISH
```

### References

- “RIATA ID” on page 305
- “RIATA START” on page 311
- *TPF System Installation Support Reference*
- “Creating the RIAT” on page 184
- Command ZRTDM in *TPF Operations*.



---

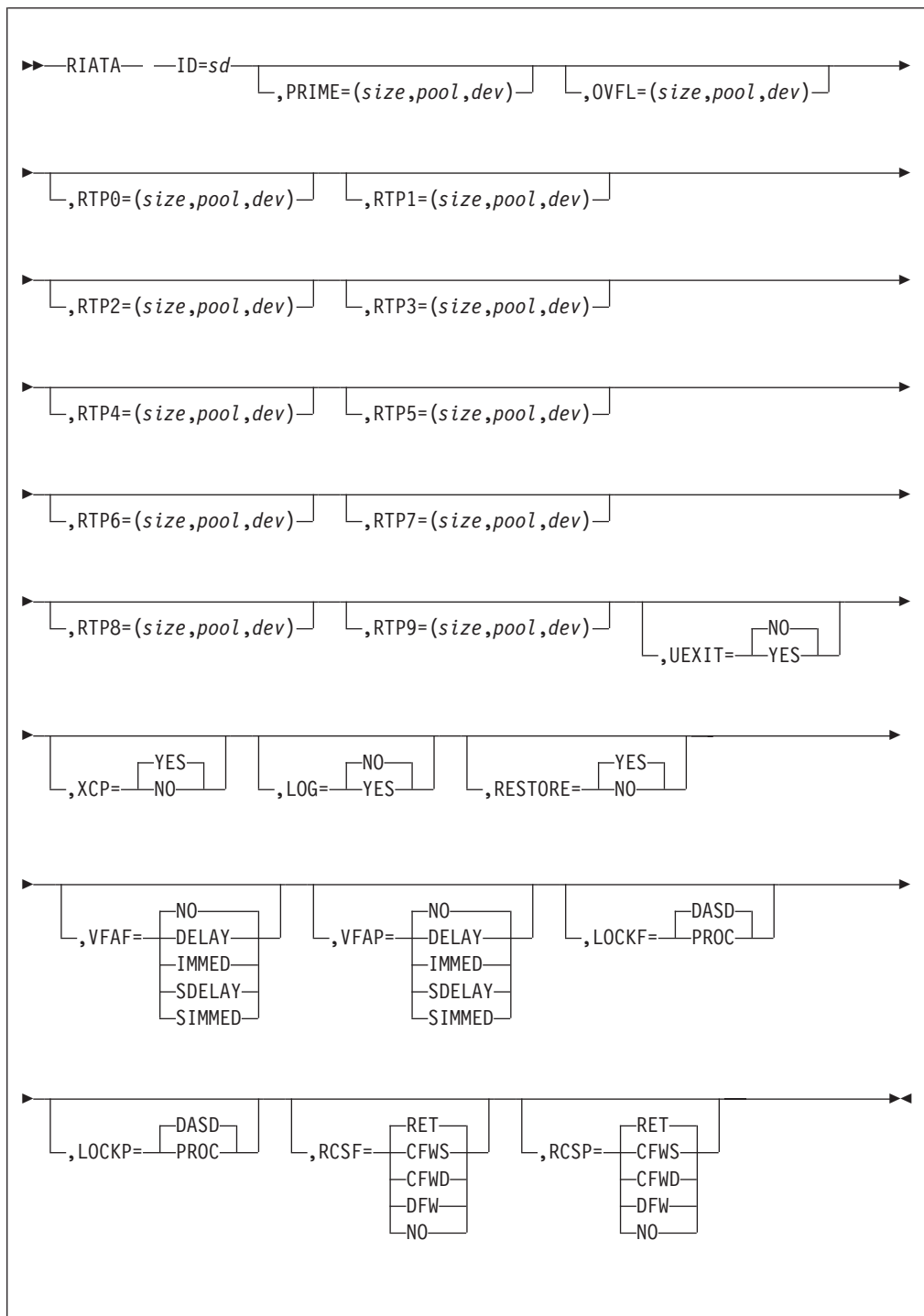
## RIATA ID

The RIATA ID macro calls are the second in a series of RIATA calls used to create entries in the record ID attribute table (RIAT).

The RIAT is created by coding a deck of RIATA calls and assembling it. The RIAT deck consists of:

1. A RIATA START call (see “RIATA START” on page 311)
2. A series of RIATA ID= calls
3. A RIATA FINISH call (see “RIATA FINISH” on page 304).

For each of the IDs to be included in the RIAT, the format is as follows:



### ID=sd

Any self-defining term, for example:

X'2100'

B'0011000011001100'

C'WW'

2134 (decimal number)

**PRIME, OVFL, RTPx**

Specify one of the following:

*size* Record size:**S** Small block**L** Large block**4** 4K block*pool* Pool duration:**ST** Short term**LT** Long term, nonduplicated**LD** Long term, duplicated**D6** Long term, duplicated, FARF6 (the size must be coded as 4)*dev* Logical device:**A** DEVA**B** DEVB**C** DEVC**D** DEVD**R** Ratio dispensing

**The default for RTP0 — RTP9 is NOT IN USE** which means the pool cannot be accessed with the GETFC macro.

PRIME and OVFL are for migration purposes only and correspond to RTP0 and RTP1. Therefore, PRIME and RTP0 cannot be coded at the same time, and OVFL and RTP1 cannot be coded at the same time.

**UEXIT**

Specify one of the following:

**NO** A user exit is not to be defined for special processing.**YES** A user exit is to be defined for special processing.**XCP**

Specify one of the following:

**YES** This record ID is to be exception recorded.**NO** This record ID is not to be exception recorded.**LOG**

Specify one of the following:

**NO** This record ID is not to be logged.**YES** This record ID is to be logged.**RESTORE**

Specify one of the following:

**YES** This record ID is to be restored.**NO** This record ID is not to be restored.

## Notes:

1. The intent of RESTORE=NO is to prevent restoration of records with a given record ID by all database utility packages that restore the database. This includes the capture/restore package, the DBR package, and any other packages providing this kind of function. In particular #PROGn and #XPRGn records should specify RESTORE=NO to save time during capture and to maintain any online modifications made subsequent to a capture.
2. If RESTORE is coded as NO, exception recording and logging can still occur so it is recommended that the LOG and XCP parameters also be coded as NO. Unexpected results can occur if RESTORE is coded as NO and either LOG or XCP are coded as YES.

## VFAF

Specify one of the following:

<b>NO</b>	This fixed record ID is not a candidate.
<b>DELAY</b>	This fixed record ID is a candidate for delayed filing.
<b>IMMED</b>	This fixed record ID is a candidate for immediate filing.
<b>SDELAY</b>	This fixed record ID is a virtual file access (VFA) synchronization candidate for delayed filing.
<b>SIMMED</b>	This fixed record ID is a VFA synchronization candidate for immediate filing.

## VFAP

Specify one of the following:

<b>NO</b>	This pool record ID is not a candidate.
<b>DELAY</b>	This pool record ID is a candidate for delayed filing.
<b>IMMED</b>	This pool record ID is a candidate for immediate filing.
<b>SDELAY</b>	This pool record ID is a VFA synchronization candidate for delayed filing.
<b>SIMMED</b>	This pool record ID is a VFA synchronization candidate for immediate filing.

## LOCKF

Specify one of the following:

<b>DASD</b>	The DASD control unit lock table maintains the lock for this fixed record type.
<b>PROC</b>	The processor's record hold table (RHT) maintains the lock for this fixed record type.

## LOCKP

Specify one of the following:

<b>DASD</b>	The DASD control unit lock table maintains the lock for this pool record type.
<b>PROC</b>	The processor's record hold table (RHT) maintains the lock for this pool record type.

## RCSF

This parameter specifies which record caching attributes apply to fixed file records with the given record ID. You can specify different options for RCSF and RCSP.

- RET** Retentive access — specifies that the record ID is to be placed on the DASD surface and cache when a file-type macro is issued.
- CFWS** Cache fast write access (simplex write) — specifies that the record ID is to be placed in the volatile control unit cache when a file-type macro is issued and the cache is available. If the cache is not available, the record is written directly to the DASD surface. A single write is issued to the prime module only.
- CFWD** Cache fast write access (duplex write) — specifies that the record ID is to be placed in the volatile control unit cache when a file-type macro is issued and the cache is available. If the cache is not available, the record is written directly to the DASD surface. A duplex write is issued to both the prime and duplicate modules.
- DFW** DASD fast write access — specifies that the record ID is to be placed in the nonvolatile storage cache when a file-type macro is issued and nonvolatile cache is available. If neither the cache nor the nonvolatile storage is available, the record is written directly to the DASD surface.
- NO** Bypass cache — specifies that the record ID is not a candidate for caching. All I/O requests for this ID result in the record being read or written directly from or to the DASD surface, bypassing caching for this record ID.

**RCSP**

This parameter specifies which record caching attributes apply to pool records with the given record ID. You can specify different options for RCSF and RCSP.

- RET** Retentive access — specifies that the record ID is to be placed on the DASD surface and cache when a file-type macro is issued.
- CFWS** Cache fast write access (simplex write) — specifies that the record ID is to be placed in the volatile control unit cache when a file-type macro is issued and the cache is available. If the cache is not available, the record is written directly to the DASD surface. A single write is issued to the prime module only.
- CFWD** Cache fast write access (duplex write) — specifies that the record ID is to be placed in the volatile control unit cache when a file-type macro is issued and the cache is available. If the cache is not available, the record is written directly to the DASD surface. A duplex write is issued to both the prime and duplicate modules.
- DFW** DASD fast write access — specifies that the record ID is to be placed in the nonvolatile storage cache when a file-type macro is issued and nonvolatile cache is available. If neither the cache nor the nonvolatile storage is available, the record is written directly to the DASD surface.
- NO** Bypass cache — specifies that the record ID is not a candidate for caching. All I/O requests for this ID result in the record being read or written directly from or to the DASD surface, thus bypassing caching for this record ID.

## Entry Requirements

The macro must be coded in the sequence described previously, the RIATA START call, followed by RIATA ID calls, followed by a RIATA FINISH call.

## RIATA ID

### Return Conditions

Output consists of either the RIAT table or MNOTEs (see *Messages (Online)*). If any of the parameters are coded incorrectly, the calls occur in the wrong order or duplicate IDs are coded. Careful understanding of the RIAT and RIATA input is required to properly interpret RIATA output.

### Programming Considerations

- The RIATA macro calls are coded in the RIAT segment, not in the stage I deck.
- The 10 pool record ratio categories are available to application programmers for more efficient utilization of disk space. Each record category should have unique attributes with respect to size, pool, and device. For example, record category 0 (RTP0) could be designated as size=L (large), pool=LT (long term, nonduplicated), and device=A (DEVA). No other record category (RTP1–RTP9) would have these same attributes. When coded, RTPx parameters must consist of a 3-element sublist of size (S, L, or 4), pool type (ST, LT, LD, or D6), and device (A, B, C, D, R).

### Examples

```
RIATA START
RIATA ID=B'0000000100000001',RTP0=(S,ST,A),RTP1=(S,LD,A),    *
      VFAP=IMMED
RIATA ID=X'102',UEXIT=YES,XCP=NO,LOG=YES
RIATA ID=5008
RIATA ID=C'AA',RTP5=(4,LD,A),RTP8=(4,LT,A),VFAP=DELAY
RIATA ID=X'C1C3',VFAP=IMMED,VFAP=IMMED
RIATA ID=C'TC',VFAP=IMMED,VFAP=IMMED,RCSF=CFWD,RCSP=DFW
RIATA FINISH
```

### References

- “RIATA FINISH” on page 304
- “RIATA START” on page 311
- *TPF System Installation Support Reference*
- “Creating the RIAT” on page 184
- Command ZRTDM in *TPF Operations*.

## RIATA START

The RIATA START macro call is the first in a series of RIATA calls used to create entries in the record ID attribute table (RIAT).

The RIAT is created by coding a deck of RIATA calls and assembling it. The RIAT deck consists of:

1. A RIATA START call
2. A series of RIATA ID= calls (see “RIATA ID” on page 305)
3. A RIATA FINISH call (see “RIATA FINISH” on page 304).

## Format



## Entry Requirements

The macro must be coded in the sequence described previously, the RIATA START call, followed by RIATA ID calls, followed by a RIATA FINISH call.

## Return Conditions

Output consists of either the RIAT table or MNOTEs (see *Messages (Online)*). If any of the parameters are coded incorrectly, the calls occur in the wrong order or duplicate IDs are coded. Careful understanding of the RIAT and RIATA input is required to properly interpret RIATA output.

## Programming Considerations

The RIATA macro calls are coded in the RIAT segment, not in the stage I deck.

## Examples

```
RIATA START
RIATA ID=B'0000000100000001',RTP0=(S,ST,A),RTP1=(S,LD,A), *
      VFAP=IMMED
RIATA ID=X'102',UEXIT=YES,XCP=NO,LOG=YES
RIATA ID=5008
RIATA ID=C'AA',RTP5=(4,LD,A),RTP8=(4,LT,A),VFAP=DELAY
RIATA ID=X'C1C3',VFAF=IMMED,VFAP=IMMED
RIATA ID=C'TC',VFAF=IMMED,VFAP=IMMED,RCSF=CFWD,RCSP=DFW
RIATA FINISH
```

## References

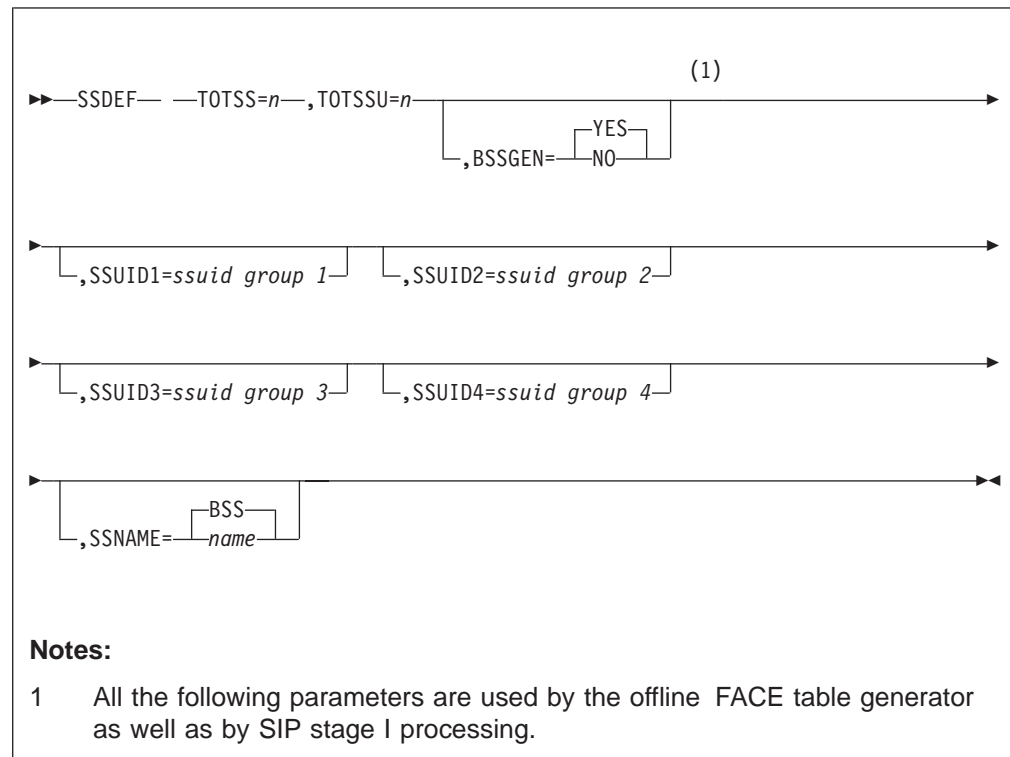
- “RIATA ID” on page 305
- “RIATA FINISH” on page 304
- *TPF System Installation Support Reference*
- “Creating the RIAT” on page 184
- Command ZRTDM in *TPF Operations*.

## SSDEF

The SSDEF macro is used to specify the definitions required for multiple database function (MDBF) support.

The SSDEF macro is only to be coded when the MDBF option is selected.

### Format



#### TOTSS=n

This parameter defines the total number of subsystems that will be supported during the life of the system. This parameter is only valid when BSSGEN=YES is coded and must be supplied. The valid range of numbers that can be supplied are 1 to 64.

**Note:** Coding this parameter incorrectly may cause insufficient storage allocation for the Dump Label Table (DLTEC).

#### TOTSSU=n

This parameter defines the maximum number of subsystem users specified for any subsystem in the complex. This parameter must be supplied and is only valid when BSSGEN=YES is coded. The valid range of numbers that can be supplied is 1 to 126 users. This parameter is used for defining storage for the data reduction program, and the generation of the Dump Label Table (DLTEC).

**Note:** Coding this parameter incorrectly may cause insufficient storage allocation for the Dump Label Table (DLTEC).

The total number of subsystems plus the total number of subsystem users must be less than or equal to 128.



**BSSGEN=**YES|NO

This parameter defines if this SIP stage I run is for the basic subsystem generation.

**Note:** If this parameter is coded as or defaulted to YES, SSNAME must be coded as or defaulted to BSS.

**SSUID1-4=***ssuid group 1-4*

Defines the 1 to 4 character alphanumeric name(s) associated with each subsystem user that will be resident in the subsystem named by the SSNAME parameter. SSUID1-SSUID4 can accept up to 24 four-character subsystem user names for a maximum total of 94 subsystem users per subsystem.

- These parameters must be coded sequentially, that is, SSUID1 must be coded before SSUID2, SSUID2 before SSUID3 and SSUID3 before SSUID4.
- If the SSUID1 parameter is omitted the subsystem user will default to the name SSU0. If SSUID2, SSUID3, or SSUID4 are omitted, no default is used.
- No duplicate names are allowed between subsystem users.

**Example:**

If 5 subsystem users will be resident in the subsystem being defined then only SSUID1 with the 5 names need be supplied.

SSUID1=(nam1,nam2,nam3,nam4,nam5) .

If 27 subsystem names are required then SSUID1 and SSUID2 must be supplied.

SSUID1=(nam1,nam2,nam3, . . . . . ,nm24) ,  
SSUID2=(nm25,nm26,nm27)

**Attention:** Every subsystem user name must be unique within the entire complex of basic subsystem and associated subsystems. In addition none of the user names may be the same as any of the subsystem names (SSNAME parameter below) or any of the application names (APLIC parameter in MSGRTA).

**SSNAME=**BSS|*name*

A 1 to 4 character alphanumeric field defining the subsystem name. When generating the basic subsystem in a MDBF environment (BSSGEN=YES), this parameter must be coded as or defaulted to BSS. SSNAME must be coded when generating a nonbasic subsystem in a MDBF environment. The first character must be alphabetic (A–Z).

## Examples

None.

## References

None.

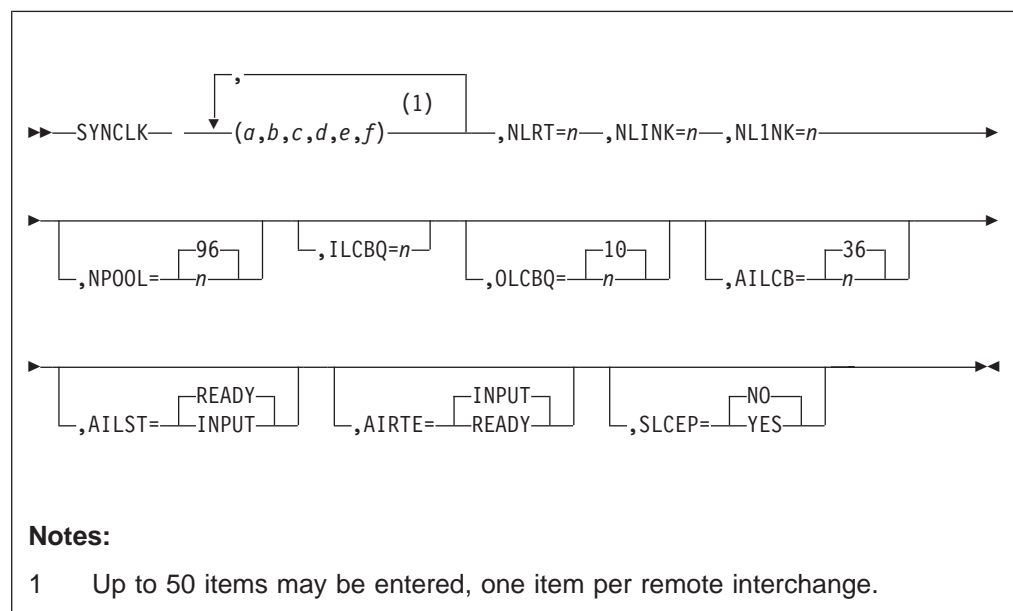
## SYNCLK

The SYNCLK macro is used to specify the synchronous link control (SLC) routing tables and features of the SLC network. It is only required when SLC line discipline is used by the generated system (SLCAI parameter in LINES macro does not equal zero).

The SYNCLK macro is only to be coded when generating a base only system or when generating the basic subsystem in a multiple database function (MDBF) environment.

See macro CTKL in *TPF System Macros* for coding instructions of the link and channel keypoints for synchronous link control support. See *TPF Non-SNA Data Communications Reference* for more information about synchronous link control.

## Format



(a,b,c,d,e,f)

These fields make up the SLC routing table. Up to 50 items may be entered, one item per remote interchange.

- a Symbolic link number (SKN) for this entry. The SKN should be submitted as a two-digit decimal number relative to the first SLC line in the system. (e.g., the first SLC line = 01, the third = 03, etc.) The SKN relative line number will be converted to an actual symbolic line number. This number must not exceed the number of SLC lines submitted in LINES (SLCAI).
- b High level exit address (HEX) associated with this remote interchange address in hex. Value is a 4-digit entry with each character not greater than 7F (maximum value = 7F7F).
- c Terminal Circuit ID (TCID) for this entry in 2-character hex. Value must not exceed 7F.

**Note:** The TCID identifies the circuit out of the high level center to which the remote interchange is attached.

- d* Pseudo-line number (PLN) for this entry. PLN is handled exactly as SKN. Decimal value must not exceed number of pseudo-lines entered in LINES (PSLNS).
- e* Interchange address (TIA) on the PLN in 2-character hex. Value must not exceed 3F.
- f* Data translate code (IND) used for the PLN. The translate code must be one of the following:
  - PSB** Padded Sabre (ALC)
  - PBT** Padded Baudot
  - ISO** International Standards Organization (standard ISO 7 bit code)
  - ESB** Extended seven bit
  - EPS** Extended padded Sabre

All of the above codes indicate that incoming messages should be passed to the input message editor (UII) package. To bypass UII, either PSBN, PBTN or ISON should be entered. If the IND field is omitted, PSB is assumed. PLN or TIA may be coded or defaulted to zero if IND field specifies that UII is to be bypassed (PSBN, PBTN, ISON, ESN or EPSN).

One table must be entered for each routing table desired. If no routing tables are required, positional parameters may be omitted.

**NLRT=*n***

Number of slots in the SLC routing table. NLRT must be the number of routing tables coded in this macro. If room for additional routing tables is to be provided for, NLRT must be larger than the number of tables that are coded.

**NLINK=*n***

Number of SLC links in the system. If pseudo-lines were entered in LINES (PLSNS), this parameter cannot equal zero.

**N1LNK=*n***

Number of SLC links with 1 AI line pair. N1LNK must be less than or equal to NLINK.

**NPOOL=96|*n***

Number of SLC input pool records for each SLC link. (Must be a multiple of 8). The fixed record type #LKIBR must be specified by the RAMFIL macro and must be a minimum of (NLINK multiplied by NPOOL).

**ILCBQ=*n***

Number of slots in the input link control block (LCB) queue. The minimum that can be coded for this parameter is 10 x the total number of SLC lines defined in the LINES macro.

**OLCBQ=10|*n***

Number of slots in the output LCB queue.

**AILCB=36|*n***

The AI CXFRC/ENQ LCB-IN level (in decimal). This is the number of 128-byte blocks below which input ENQ LCBs will be discarded. The number specified must be considerably less than N128 specified in the CORREQ macro.

**AILST=READY|INPUT**

Defines the routing of input message blocks from CCP to OPZERO as via the INPUT or READY list.

## SYNCLK

### **AIRTE=INPUT|READY**

Defines the routing of last or only type A input message block from CMR1 to CMR3. If the AILST parameter equals INPUT, AIRTE is not required and if coded will be ignored.

### **SLCEP=NO|YES**

Defines whether or not SLC support for the communication controller in EP mode is desired.

## Examples

This example defines two SLC routing table entries. Both lines use the padded Sabre (ALC) line control. It is assumed that the LINES macro has two SLC and two pseudo SLC lines defined. Two additional slots will be defined for the routing table. All other nondefined fields will default.

```
SYNCLK (01,7070,70,01,01,PSB),      X
      (02,7171,71,02,02,PSB),      X
      NLRT=4,NLINK=2,N1LNK=2,SLCEP=YES
```

## References

None.

---

## UFTEND

This macro must be coded after the UFTFTI macro. It contains no parameters and is used by the assembler to signal the end of the parameters coded on the UFTFTI macro.

## UFTFTI

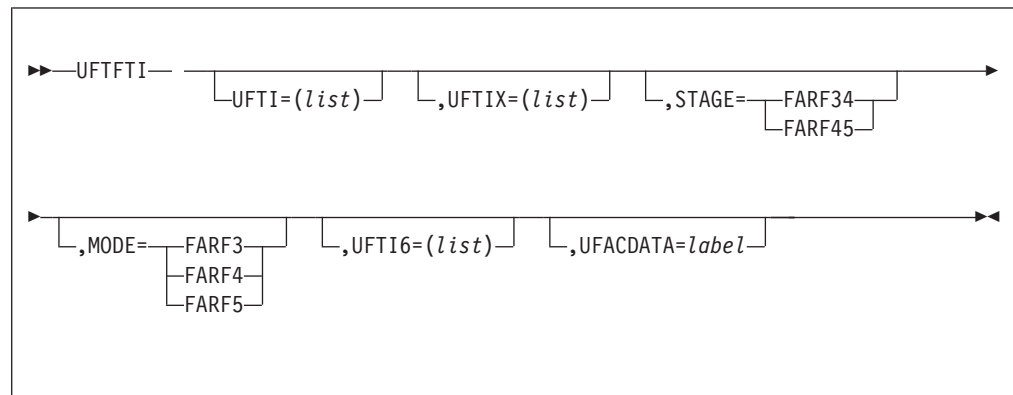
The UFTFTI macro is used by the offline FACE table generation program to define UFT/FTI characteristics for use by FARF4, FARF5, and FARF6. It is not necessary to code UFTFTI statements for a FARF3-only FACE table. UFTFTI also defines the FARF migration stage, as well as the default address dispense mode, and provides a means to attach a single user data structure related to the entire FACE table.

For each UFT, the length (in bits) of the FTI is defined. Each UFT/FTI-length combination is coded as an ordered pair. UFT/FTI pairs defined for FARF6 are independent from UFT/FTI pairs defined for FARF4 and FARF5.

There are two FARF stages: FARF3/4 and FARF4/5. FARF3/4 allows both FARF3 and FARF4 address to coexist in the database. FARF4/5 allows both FARF4 and FARF5 address to coexist in the database. FARF6 addresses are dispensed in either stage and are independent of dispense mode.

There are two possible default address dispense modes: FARF3 and FARF4. The allowable subsets of address dispense modes for a particular FACE table directly correspond to the address formats allowed for the table's migration stage.

## Format



### UFTI, UFTIX=(list)

Specifies a list of UFT-value/FTI-length combinations expressed as ordered pairs.

Each UFT is expressed as a decimal number from 0 to 63. The FTI-length is the length in bits of the FTI field for any address using this particular UFT (fixed or pools, FARF4 or FARF5). Each FTI length is expressed as a decimal number

- from 1 to 23, if the UFT is to be used for ANY FARF4 or FARF5 addresses or
- from 1 to 25, if the UFT is exclusively dedicated to FARF5 use.

If the dispensing mode is FARF34 and the database is only to be defined in terms of FARF3 addresses, it is not necessary to code the UFTI parameter.

A total of 64 UFT/FTI pairs can be coded. Because the assembler has a limit of 255 characters in a sublist (about 32 UFT/FTI pairs), both UFTI and UFTIX are provided to avoid assembly errors. Code UFT/FTI pairs on the UFTI parameter first until you reach the 255-character limit and then use the UFTIX parameter.

The following is an example of a valid UFT/FTI length combination.

UFTFTI     UFTI=(0,10),(1,2),(2,25)

This UFTI specification specifies three UFTs 0, 1, and 2, and their associated FTI lengths. There are  $2^{10}$  or 1024 FTIs associated with UFT 0,  $2^2$  or 4 FTIs with UFT 1, and  $2^{25}$  or 33,554,432 FTIs with UFT 2. Moreover the specification defines the number of records that can be defined with each UFT and FTI, since six bits of an address are devoted to the UFT and the number specified by UFTI= indicates the number of bits devoted to FTIs. For UFTI=(0,10) we have:

32 bits - (6 bits UFT + 10 bits FTI)  
= 32 - 16 = 16 bits FARF5 ordinal numbers

16 bits - 2 bits for FARF4 = 14 bits FARF4 ordinal numbers

16 bits for FARF5 addresses and 14 bits for FARF4 addresses (removing the FARF4 control bits from the total).

#### **UFTI6=(uft,fti)**

Specifies a list of UFT-value/FTI-length combinations expressed as ordered pairs. These are the UFT/FTI pairs that are used for FARF6 addressing only. These pairs are independent of the pairs used for FARF4 and FARF5 addressing.

Each UFT is expressed as a decimal number from 1 to 65535. The FTI-length is the length, in bits, of the FTI field for a FARF6 address using this particular UFT. Each FTI length is expressed as a decimal number from 8 to 24. A UFT value of 0 is illegal. To reduce the amount of storage required by the FACE table, define the lowest UFT values first.

#### **STAGE**

Specify one of the following:

- |               |  |
|---------------|--|
| <b>FARF34</b> | A migration stage from FARF3 to FARF4. A FACE table that supports both FARF3 and FARF4 address dispense modes is generated. This is the default. |
| <b>FARF45</b> | A migration stage from FARF4 to FARF5. A FACE table that supports both FARF4 and FARF5 address dispense modes is generated.                      |

**Note:** FARF6 addresses are dispensed in both FARF34 and FARF45 address modes.

#### **MODE=FARF3|FARF4|FARF5**

The MODE parameter defines the default address dispense mode for dispensing FARF addresses (for fixed and pool record types). This attribute can be changed online by command ZMODE. If MODE is not coded, the default dispensing mode in STAGE=FARF34 is FARF3 and in STAGE=FARF45 it is FARF4.

#### **UFACDATA=label**

The UFACDATA parameter attaches user data at the FACE table level. The label must be a valid entry point in the user data, as it results in a VCON in the FACE table. The VCON is resolved by the MVS linkage editor. The effect of this mechanism is to extend the information provided in the FACE table by allowing user specified data to be associated with a particular FACE table.

The CTSD, CONK, and IRCCR symbols are reserved and must not be used.

**UFTFTI**

## **Examples**

None.

## **References**

None.

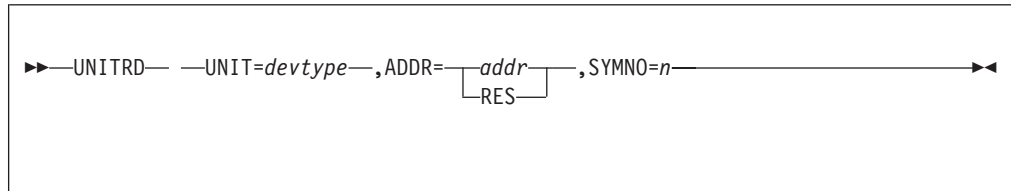


## UNITRD

The UNITRD macro is used to specify the characteristics and requirements of unit record equipment.

There must be a UNITRD macro for each unit record device in your system. Printers and card readers are supported as unit record devices. The maximum number of unit record devices allowed is eight.

## Format



### UNIT=*devtype*

The type of unit record device. Specify one of the following:

#### 1403|3211

The type of printer.

**Note:** The 4248 Impact Line Printer, Model 2, has the ability to act in 3211 emulation mode.

#### 2540R|3505

The type of card reader.

### ADDR

Specify one of the following:

*addr* The physical address of the device. It must be 3 hex digits with the first digit being 0.

**RES** No address but reserves a slot for a future device.

**Note:** If the address coded is the same as the APRNT parameter of the CONFIG macro the device will be shared with the IPL and dump program.

### SYMNO=*n*

Specifies the symbolic number assigned to the device: (1–7) for printers, or (1–3) for readers.

**Note:** Unique symbolic numbers must be coded for devices of the same type.

## Examples

The following specifies a 1403 printer with a unit address of 00E and symbolic number of 3.

```
UNITRD ADDR=00E,SYMNO=3,UNIT=1403
```

## References

None.

## USEREQ

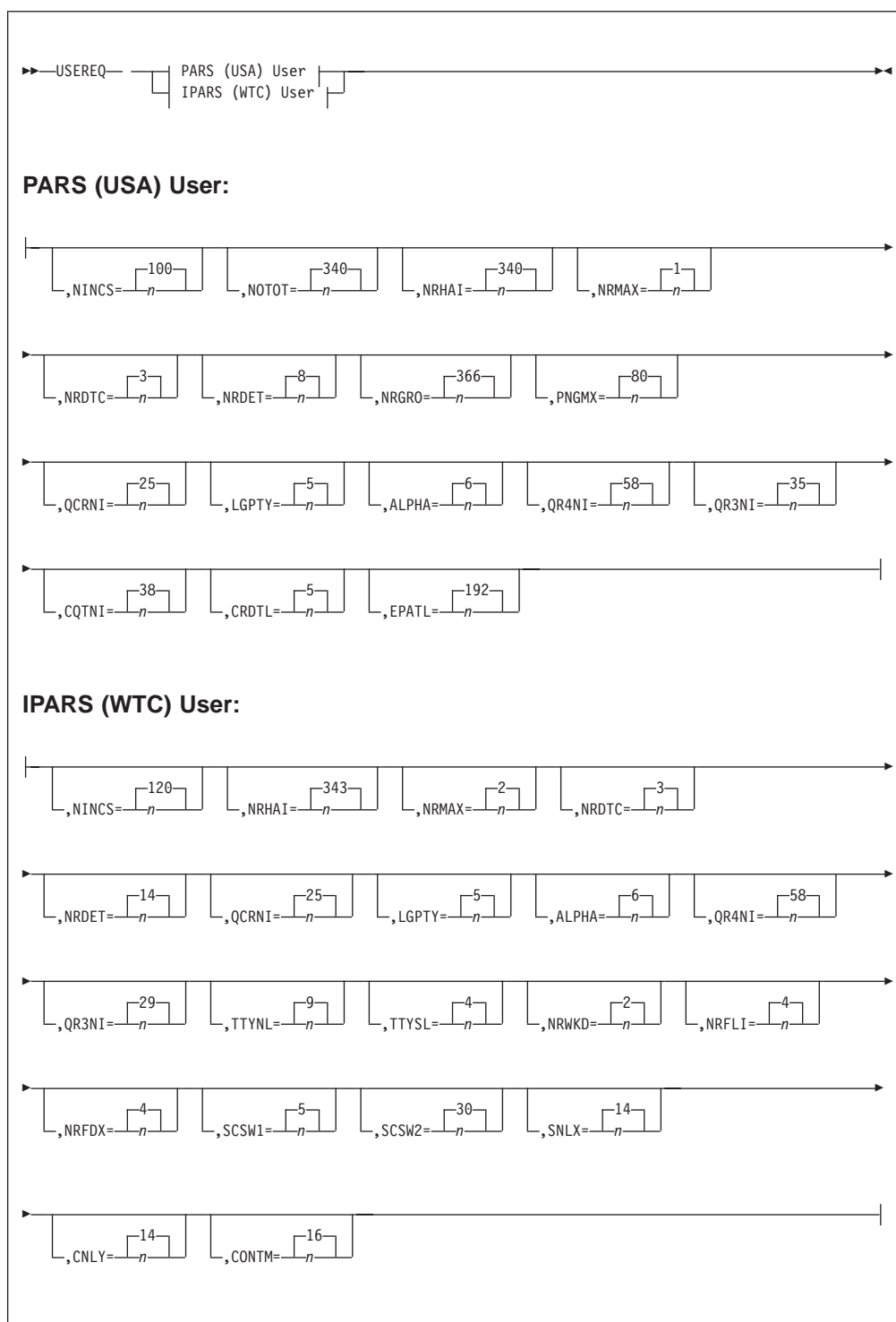
---

### USEREQ

The USEREQ macro is used to specify Airlines Reservation user dependent variables. These values will be passed to the SYSEQC macro as symbolic equates which are used by user application programs.

This macro is not required.

## Format

**NINCS=D|n**

Number of cities serviced.

**NOTOT=D|n**

OA Trouble range days FIT records kept.

**NRHAI=D|n**

Total HA inventory range.

## USEREQ

**NRMAX=D|n**

Number of NAR records per day.

**NRDTC=D|n**

Maximum number of data changes per flight.

**NRDET=D|n**

Number of days in the detail period. For IPARS this must be a number of whole weeks.

**NRGRO=D|n**

Number of days in the gross period.

**PNGMX=D|n**

PNIG Maximum of details and items.

**QCRNI=D|n**

Number of items in QCR.

**LGPTY=D|n**

Minimum number for large party.

**ALPHA=D|n**

Number of alpha groups.

**QR4NI=D|n**

Number of items in MSG ORD.

**QR3NI=D|n**

Number of items in PNR ORD.

**CQTNI=D|n**

Number of items in CQT.

**CRDTL=D|n**

Number of seats sold to create IND.

**EPATL=D|n**

Number of records in error table.

**TTYNL=D|n**

TTY 'need' limit.

**TTYSL=D|n**

TTY 'sell' limit.

**NRWKD=D|n**

Number of weeks in detail period.

**NRFLI=D|n**

Maximum number of FLI records.

**NRFDX=D|n**

Maximum number of FDX records.

**SCSW1=D|n**

Time allowed for detailing.

**SCSW2=D|n**

Time allowed for detailing for Schedule Change.

**SNLX=D|n**

Sales notification recalculation percentage.

**CNLY=D|n**

Cancellation recalculation percentage.

**CONTM=D|n**

Maximum interval for connecting services.

Where D = default value specified in either USA or WTC description. (See preceding prototype for default values.)

## Examples

None.

## References

None.

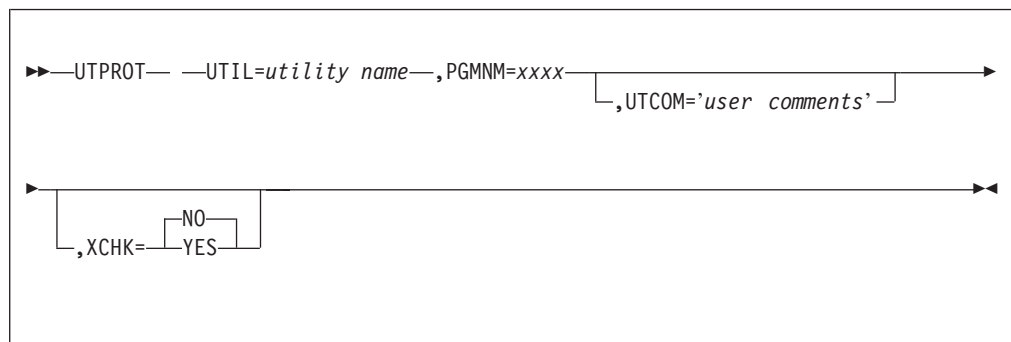
## UTPROT

The UTPROT macro is used to specify names to the table of utility interface programs which are to be entered during a utility processor ownership assignment or release. This macro may be submitted for fifty utilities.

UTPROT is required in a loosely coupled environment when the system being generated is a base only system or the basic subsystem of a multiple database function (MDBF) environment.

This macro will create the macro UIPROT in stage II to be used exclusively by program CNPO.

## Format



### UTIL=*utility name*

Identifies the system utility to be controlled via the processor resource ownership table by subsystem user, for example recoup, capture/restore, pool support. This is a 4-character unique utility identifier to be used with the utility interface program table in the program segment CNPO.

### PGMNM=xxxx

Specifies the utility interface segment to be entered during a utility processor ownership assignment or release. xxxx is a 4-character interface segment. It is used to validate and set up tables for the utility support as desired.

### UTCOM='user comments'

This parameter is optional and is provided for user comments. The user's comments must be enclosed in quotes and cannot be more than 35 characters in length.

### XCHK=NO|YES

Specifies whether the utility interface segments are to be checked with the real time program tables created by SIP (SPPGML). This parameter should only be coded on the first UTPROT macro.

#### Notes:

1. If the user specifies YES, all programs defined in this macro will be checked with the real time program tables. This will substantially increase the time required for generation. If this parameter is specified as NO, the user must be certain that the utility interface segments coded in this macro are also included in the real time program tables in the program list macro (SPPGML).
2. Because the validation of input program names will **substantially increase** the time of the system generation, the default is no.

## Examples

The following example specifies three system utilities to be controlled via the program resource ownership table. The respective utility interface segments to be entered during a utility processor ownership assignment or release have also been coded. The XCHK parameter has been coded NO since the validation of program names will SUBSTANTIALLY INCREASE the time of the system generation.

```
UTPROT  UTIL=CAP0,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY0',XCHK=NO
UTPROT  UTIL=CAP1,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY1'
UTPROT  UTIL=CAP2,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY2'
```

## References

None.

**UTPROT**



---

## Stage II: Executing the Job Stream

This job stream must be executed in the same sequence that it was created, as output from stage I, because of dependencies between many jobs and job steps. Since execution of stage II involves much machine processing, it is recommended that a large amount of core (REGION) be available and other activities be limited because of I/O requirements. Actual run time will vary depending upon the stage I parameters that were used to create the stage II job stream.

---

### Analyzing Errors

Errors that occur during stage I may stop creation of the job-stream and descriptive messages will be printed indicating where the errors occurred. See *Messages (System Error and Offline)* for the system generation error messages. After the errors are corrected, stage I must be rerun.

Any errors that occur during stage II processing will require a similar approach. If these errors resulted from stage I input, stage I must be rerun. Any stage II errors can be analyzed by using the information listed in Appendix C, "SIP Stage II Job Summary" on page 473, which shows a listing of stage II jobs and steps. Once again, errors must be corrected and stage I rerun prior to executing stage II, except for possible hardware or overload type of errors.

---

### SIP Report

Stage I produces a detailed listing of all the input parameters and corresponding calculations. This report details the final generation results based on input parameters specified. In addition, any variable calculations based on the input parameters will be shown. It is strongly recommended that this report be analyzed thoroughly before attempting to execute the stage II process. The SIP report is divided into the following sections:

1. Description of optional support
2. Detailed description of user macros.

### Description of Optional Support

This section will detail the different optional support features that will be included in the system generation. In addition, a sub-list defining any of the user macros that were not coded will be printed. The user should review both of these lists to insure that all of the required optional support needed for a successful operating system has been defined.

### Detailed Description of User Macros

This section spells out all the user inputs and any variable calculations based on that input. The user should check the results reported for each macro against the inputs supplied to that macro to insure the correct results have been attained.

---

### Revisions to Job Stream

Use care when attempting to modify the stage II job stream because of the interdependencies between various jobs. For example, modifying any of the CIMR components causes a change to the Image Pointer Record (CTKX).

---

## Types of Output

The following summarizes the various types of output that result from stage II. See Appendix C, “SIP Stage II Job Summary” on page 473 for the order and complete description of the stage II job stream.

- Source code for keypoints, table label records, macros, and others created by SIP will exist in ACP.SYSRCE.RELv and ACP.SYMACRO.RELv data sets, respectively.
- Because stage II uses a multiple assembler/compiler (see *TPF System Installation Support Reference* for more information about the multiple assembly/compilation program), the assembly and compilation listings for all error-free programs will be on a tape when the LISTAPE parameter of the GENSIP macro is used. You can then request a specific listing as provided in *TPF System Installation Support Reference*.
- The ANTDEF input statements generated by SIP for the OSTG program will be put into the data set named by the ANTPDS parameter on the GENSIP macro. (See *TPF ACF/SNA Network Generation*.)
- Applicable object and load modules for all segments contained in SPPGML will exist in respective object and load module libraries. On a WTC system this will include the real-time program segments listed in SPIPML and the offline components listed in SPWTC.
- A list for this release named PARSV and located in data set ACP.SYSRCE.RELv is generated. This list is then referenced in the load deck described previously. It will include all segments listed in SPPGML as real-time segments, except for those segments that need be loaded only once. For WTC systems, a second list will be generated containing all the IPARS real-time program segments as defined by SIP internal macro SPIPML. IBMPAL macros and any user allocator decks.
- An allocator table will exist in data set ACP.SALTBL.RELv.
- A load deck is generated and listed during stage II as member LOADDECK of data set ACP.SYSRCE.RELv. This deck is run against ALDR to create the loader general file (LGF).
- Loader general file DASD will be ready for executing the load deck.

---

## Post-SIP Stage II

After successfully completing FACE table generation SIP stage I and II, you should complete the implementation of your system. This may involve executing the functionally dependent offline packages such as: pool directory generation for DASD devices, communication controller loading, preparation of the online devices (ICKDSF and formatting), pilot tape creations, and others.

---

## Other Generation Processes Executed During Stage II

### GTSZ Getsize

GTSZ is an offline C program that obtains the size of components to be placed in the core image restart area (CIMR).

C\$GTSZ is a C header file created as part of SIP stage II. It contains a list of CIMR components, with version codes, which GTSZ is compiled against. GTSZ gets the sizes of the components listed by C\$GTSZ.

Output from GTSZ is segment SKGTSZ, a list of macro language global variables, one for each CIMR component, set to the size in bytes of the corresponding CIMR component. For example, a sample list looks like the following:

&XAACPL	SETA	13664	ACPLxx
&XAICDF	SETA	80972	ICDFxx
&XAIPAT	SETA	339108	IPATxx
&XAIPLB	SETA	73032	IPLBxx
&XARIAT	SETA	23000	RIATxx
&XASIGT	SETA	1780	SIGTxx

These values are used in creating CTKX. A 30% expansion factor is added when CTKX is created.

## Running GTSZ

The recommended way to run GTSZ is assembling a SIP stage I deck and executing the output in SIP stage II. Users who want to run GTSZ independent of SIP because of customization should assemble a SIP stage I deck to get the initial JCL necessary to compile, link and run GTSZ, and get header file C\$GTSZ. Users then carefully update header file C\$GTSZ to include the CIMR component names and version codes they require. Each component must be added to the appropriate array: additional segments to be searched for in an object module data set go into the *objname* array while additional segments to search for in link module data sets go into the *loadname* array. The order of segments in the arrays is irrelevant. Users should be familiar with the C programming language before doing this. The following is a sample C\$GTSZ:

```

/*****
/* THIS HEADER WAS PRODUCED BY SIP SEGMENT SKSYPS */
/*****
char *objname[] = {
    "ACPLxx",
    "ICDFxx",
    "IPLBxx",
    "SIGTxx",
    "RIATxx",
    "IPATxx"
};

#define OBJNUM (sizeof(objname)/sizeof(char *))

char *loadname[] = {
    "FCTBxx",
    "CPS0xx"
};

#define LOADNUM (sizeof(loadname)/sizeof(char *))

```

- The array *objname* is a list of CIMR components that are searched for in an object module data set.
- The array *loadname* is a list of CIMR components that are searched for in a link module data set.

Once C\$GTSZ has been updated and put into data set ACP.SYMACRO.RELxx.yyy (yyy = BSS or other subsystem names) JCL similar to that below should be run:

```

//J1A1E EXEC EDCCLG,INFILE='ACP.CSRCE.OL.RELxx(GTSZ40)'
//COMPILE.SYSLIB DD DSN=&VSCCHD&CVER&EDCHDRS,DISP=SHR
//              DD DSN=ACP.SYMACRO.RELxx.yyy,DISP=SHR
//GO.SYSUDUMP DD SYSOUT=A
//GO.OUTFILE DD DSN=ACP.OBJ.RELxx.yyy(SKGTSZ),DISP=OLD
//GO.OBJDSN DD DSN=ACP.OBJ.RELxx.yyy,DISP=SHR
//GO.LNKDSN1 DD DSN=ACP.LINK.RELxx.yyy,DISP=SHR
/*

```

- The JCL above compiles, links and executes the file referred to by the INFILE parameter.

- Header files are searched for in the data sets pointed to by the COMPILE.SYSLIB DD name.
- The GO.SYSUDUMP DD name specifies the file for GTSZ error messages.
- The GO.OUTPUT DD name specifies where the output file, SKGTSZ, is to be written.
- GTSZ searches data sets specified by GO.OBJDSN when trying to find the size of object modules. Multiple data sets can be concatenated to GO.OBJDSN. The search order is the same as the concatenation order.
- GO.LNKDSN1 points to the first link module data set scanned by GTSZ when trying to find the sizes of link modules. Multiple data sets can NOT be concatenated to this DD name. Users may use GO.LNKDSN2 through GO.LNKDSN4 for additional link module data sets. GTSZ searches first in LNKDSN1 then LNKDSN2 and so on.

Once GTSZ is run and SKGTSZ is created, segments which rely on SKGTSZ (such as CTKX) should be reassembled and reloaded if necessary.

---

## Part 4. Generating the TPF Communications Network



---

## Non-SNA Communications

The system communication keypoint (SCK) and processor keypoint status table (PKST) record generation, hereafter called SCK gen, should be run for all communication lines connected to TPF via an EP or PEP communications controller and for 3270 locals. In addition to building the two types of records described below, it also, automatically, builds two System Communication Keypoint Records (SCK) for the prime and RO (receive only) CRAS (Computer Room Agent Set) terminals. Therefore, in the case of a system whose network consists solely of communication lines connected to TPF via a communication controller in NCP mode, with no 3270 locals, this generation need not be run. However, in that case, those two SCK records for the prime and RO CRAS will not be built automatically and should be created by hand, via the System Test Compiler (STC) and pilot tape load.

The SCK gen Phase I input consists of a set of user macros that describe the non-SNA communication network. The output of Phase I is a sequential file containing the network information in STC input format. The SCK gen Phase II consists of running STC with that sequential file as input. The output of Phase II is a non-SNA Communication pilot tape that contains the two record types described below. This pilot tape must be loaded later to the online system via the TPF Data Loader.

The two record types that are produced by the SCK gen are:

**SCK** System Communication Keypoints (one record/line)

These contain the status of each communication line attached to the system. They are used at restart time to build core resident communication tables:

Line Status Table (LSTB)

TI Status Table (TITB)

Symbolic Line Status Table (SLST)

**PKST** Communication Control Unit Keypoints

These contain the status of each control unit on the system.

---

## SCK Generation Preparation

Before generating the non-SNA communications pilot, the user must have completed designing his communications network. *TPF Concepts and Structures* offers an overview of network design considerations, as does "Data Communications Support" on page 123. Information entered in SIP macros is passed to the SCK macros via the SYGLBK and SYSETK members in the SYMACRO data set created during SIP execution. This ensures basic integrity between the two generations (e.g., binary synchronous station names and polling/selection sequences are the same in both SIP and SCK generation). SIP stage II must be successfully completed prior to SCK generation.

### SCK Generation Phase I

There are three SCK generation user macros:

**SKLNG**

**PKSTG** control unit

## SENDG

The macros must be used together according to the following rules:

1. The following set of macros is required, in the order specified, for each SLN connected to a 3705/EP or 3745/PEP and for each 3270 local terminal:  
SKLNG  
SENDG
2. One PKSTG macro is required for each 3705/EP and 3745/PEP control unit. The PKSTG macros must follow all the other Phase I input.
3. The very first SENDG macro in the Phase I input stream must contain STC option parameters. The very last macro in the Phase I input stream must be a SENDG END. This will initiate error analysis if it is necessary. All SENDGs other than the first and last should have no parameters included.
4. It is advantageous to code the SKLNG/SENDG macro sets in SLN sequence, and the PKSTG macros in control unit number sequence. If they are sequential, the Phase I SCK gen will perform sequence checking to ensure that all SCK and PKST records are initialized. Otherwise the user must ensure that all records are initialized. Additionally, the macro sets for 3270 local terminals must be sequential by SLN within control unit number.
5. Symbolic Line Numbers are assigned to non-SNA communications line in the following order:

Line Type	SLN
RO (Receiver Only CRAS)	00 (Always)
PRC (Prime CRAS)	01 (Always)
SLC	.
BSC	.
3270 local	.

During Phase I the user coded SCK/PKST macro statements are analyzed for errors. Some errors will prevent SCK/PKST record generation from continuing but in most cases errors will be flagged and processing will continue, using default values wherever possible, to produce an assembly listing and STC input for Phase II. If errors occur in Phase I, they must be resolved according to the instructions for each error as described in *Messages (System Error and Offline)* before proceeding to Phase II.

## SCK Generation Phase II

Phase II is a straight forward STC run. Before executing STC, the unit test disk, DISK01, should\*\*\* be loaded with SDMF and a DRIL file containing at least the following macros:

<b>SCKDS</b>	Data record description for SCK
<b>CPTIC</b>	Data record description for PKST
<b>DMPPT</b>	Data record description for PDUMP

Output from Phase I can then be used as input (\*CARDIN) to STC to produce a non-SNA communications pilot tape (\*\*TAPE1). This tape is later input to the data loader, via a command, to initialize the communications keypoint records on file.

**Note:**



- \* DDNAME given to the DCB describing the input file.
- \*\* DDNAME given to the DCB defining the STC output file.
- \*\*\* Loading of the SDMF/DRIL disk can be accomplished at the time the SCK Pilot is created by coding the parameter 'LOAD=YES' in the SENDG macro. The user must insure that the DD statement for the SDMF/DRIL tape is included in the STC JCL.

---

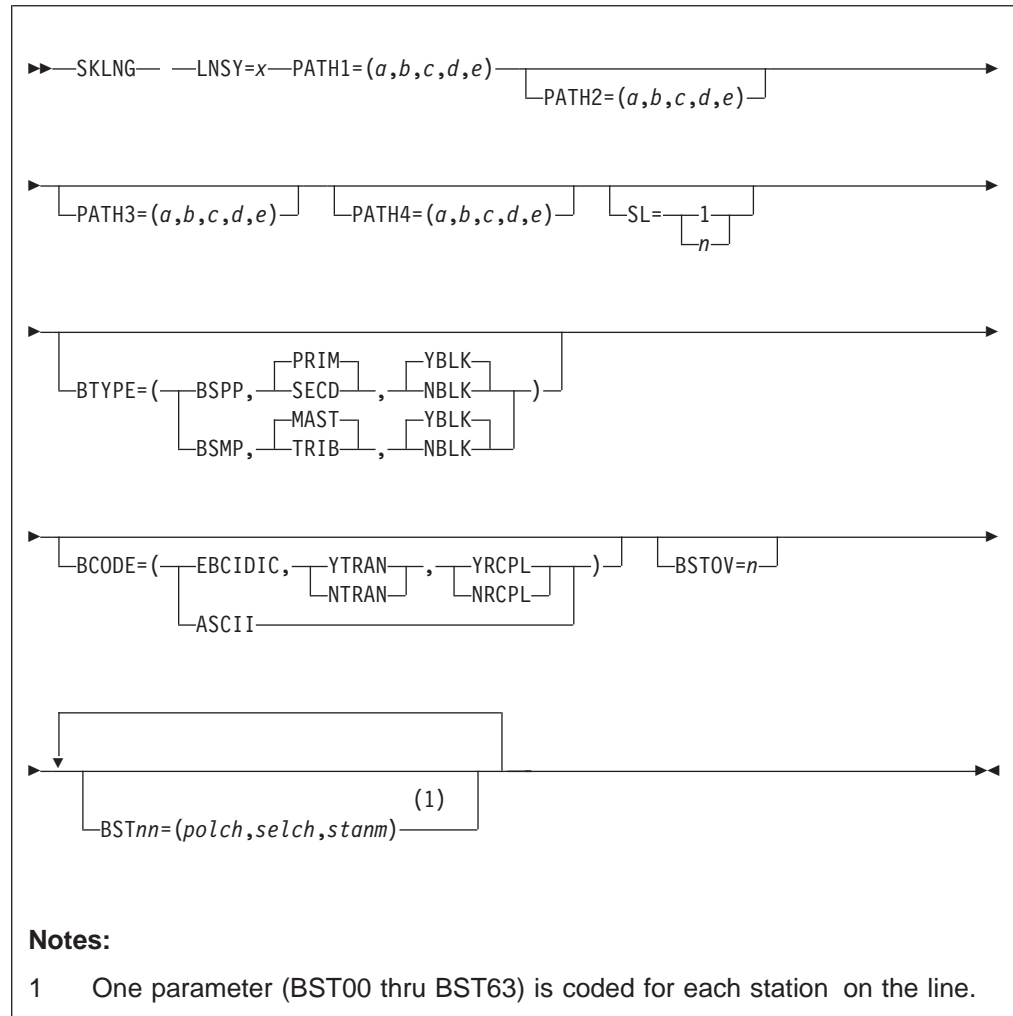
## Specifying the Communications Network

This section provides details for specifying each of the SCK/PKST generation macros. The macros are described in the order in which they must be coded and their function and parameters are summarized.

**SKLNG**

The SKLNG macro is used to specify information required to initialize the SCK. For all line types it defines the symbolic line number and the physical paths through which the line may be attached to the system. Other parameters should be coded depending on the line type.

## Format



The hexadecimal symbolic line number. The SCK macros should be coded in line number sequence, and this parameter will be used for sequence checking. Symbolic line numbers 0 and 1 should never be coded, because they are reserved for the RO and PRC. The first non-CRAS line number (for example, SLC, BSC, etc.) must be equal to that specified in the MINLS parameter of the SIP LINES macro. The symbolic line number is used to determine the line type by reference to the SIP output. If the user is in any doubt as to which line numbers represent which line types, he should refer to a listing of SYSETK which defines line number ranges for each line type.

For local 3270, each 3277 or 3284/3286 terminal is represented by a symbolic line number. These line numbers must be consecutive and therefore must be specified by consecutive SKLNG macros.

**PATH1**

Specifies the first or only physical path through which the line is attached to the system. For local 3270 lines, only one path can be defined.

- a* One to three hex digits defining the subchannel address of the path.
- b* A decimal symbolic control unit number to which the line *a*, is attached. The control unit type is determined from the control unit number by reference to the SIP output. If the user has any doubt as to which control unit numbers represent which control unit types he should refer to a listing of SYSETK which defines control unit number ranges for EP control units. This parameter should not be coded for local 3270s.
- c* Code L
- d* Indicates the switching status for each path.
  - M** Manually switchable (requiring operator command).
  - A** Automatically switchable (e.g., TPF automatically activates an alternate path or subchannel when the line error threshold is exceeded). Default is M.
- e* Indicates the type of path (must be P for local 3270's)
  - P** Prime path (active subchannel for this line).
  - A** Alternate path (back up or alternate subchannel). Default is A. If only one path is specified it must be the prime path.

**PATH2**

These parameters specify up to three more physical paths.

**PATH3**

They have the same format as the PATH1 parameter.

**PATH4**

The number of paths specified must not exceed the value defined by the NPATH parameter of the SIP NETWK macro.

**SL=1|n**

For a BSC line: this parameter defines the number of messages that will be sent before line turnaround occurs and messages will be received.

**BTYPE**

For a BSC line: this parameter defines the type of line and station.

**BSPP** Line is a point to point line.

**BSMP** Line is a multipoint line.

**PRIM** Station is the primary station. (TPF wins when contention occurs in bidding for the line). This parameter is only valid for BSC point to point lines.

**SECD** Station is the secondary station (TPF yields to the primary station when contention occurs). This parameter is only valid for BSC point to point lines.

**MAST** Defines type of station for a multipoint line. MAST is the only valid parameter. (TPF is responsible for polling the line.) This parameter is only valid for BSC multipoint lines.

**YBLK** Line uses blocked message transmission.

**NBLK** Line does not have blocking.

### BCODE

For a BSC line: this parameter defines type of code used on line.

**EBCDIC** Line uses EBCDIC codes.

**ASCII** Line uses ASCII codes.

**YTRAN** Line has data transparency. This parameter is valid only if EBCDIC is specified, since TPF only supports USASCII in nontransparent mode.

**NTRAN** Line does not have data transparency. This parameter is valid only if EBCDIC is specified, since TPF only supports USASCII in nontransparent mode.

**YRCPL** Line sends and receives RCPL as a message header. This parameters is valid only if EBCDIC is specified. YRCPL is valid only if YTRAN is also specified since transparency is required to transmit binary data in the RCPL.

**NRCPL** Line does not use RCPLs. This parameters is valid only if EBCDIC is specified.

Default values depend upon the type of BSC line. If the line is a point to point line (BTYPE=BSPP), then the defaults are EBCDIC, YTRAN, YRCPL. If the line is a multipoint line (BTYPE=BSMP), then the defaults are EBCDIC, NTRAN, NRCPL.

### BSTOV=*n*

For multipoint BSC lines: this parameter defines the time-out value in seconds.

For Master stations, this value is the slow poll interval. For tributary stations, this value is the time-out value. If no poll is received and time out value elapses, then messages on the output queue are returned to originator.

The value can be from 0 to 255. If the station is a tributary station (BTYPE=(BSMP,TRIB)), then a value of 0 can not be specified. The default value depends upon the type of station. If the station is a master (BTYPE=(BSMP,MAST)), then the default value is 15. If the station is a tributary (BTYPE=(BSMP,TRIB)), then the default is 5.

### BST00...

For BSC line: these parameters define entries in the TITB table for the line.

One parameter (BST00 thru BST63) is coded for each station on the line. The numerical part of the BSTnn parameter is the station number and relates to the STANO parameter in the BSNCT SIP macro.

*polch* This subparameter defines the station poll characters.

*selch* This subparameter defines the station select characters.

*stanm* This subparameter defines the station name.

For a point to point BSC line, only BST00 is valid. The polch and selch parameters have no meaning and should not be specified. BST00 should be specified when station name message routing is going to be used on a point to point line.

For a tributary station on a multipoint line, only BST00 is valid.

For a master station on a multipoint line, all BST parameters are valid. The BST parameters should have consecutive numbers starting at 00.

The polch and selch subparameters are used to generate pointers to entries in the BSAT table. The polch and sel characters must match an entry in the BSAT table.

The BST parameter numbers are used to create pointers to entries in the SNCT table (generated by SIP). Each BST parameter generates a TITB entry. The TITB entry points to the entry in the SNCT table which has a matching line and station number. Note, the BST parameters specified here are numbered in decimal and the station numbers specified in SIP (BSNCT macro, STANO parameter) are numbered in hexadecimal.

## **Examples**

None.

## **References**

None.

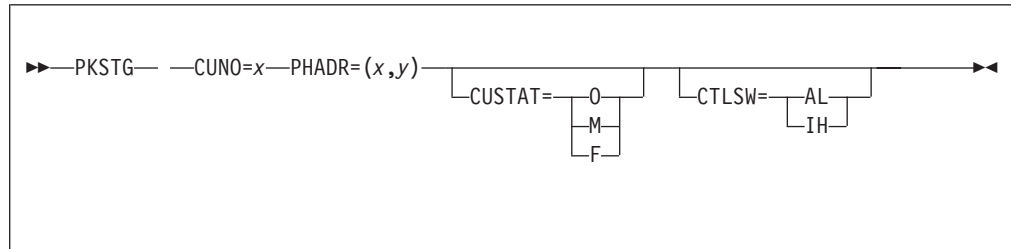
## PKSTG

### PKSTG

One PKSTG macro should be coded for each 3705/EP and 3745/PEP control unit. The PKSTG macros should be at the end of the input stream and should be in control unit number sequence.

Each PKSTG macro generates a PKST record. It defines the control status, the control unit type and the subchannel range of an EP control unit.

### Format



#### CUNO=x

The symbolic control unit number in decimal. PKSTG macros should be coded in ascending sequence by control unit number starting with zero.

#### PHADR=(x,y)

For an EP or PEP control unit: This term indicates the range of subchannel addresses. x and y are two digit hexadecimal numbers to represent the minimum and maximum data subchannel address respectively.

#### CUSTAT=O|M|F

This parameter defines the initial status of a control unit. The default is off line.

#### CTLSW=AL|IH

This parameter sets the 2703 idle status.

**AL** The 2703 is not idle.

**IH** The 2703 is idle.

### Examples

None.

### References

None.

## SENDG

The SENDG macro diagnoses incompatibilities in SKLNG macro calls, and generates an SCK/PKST record.

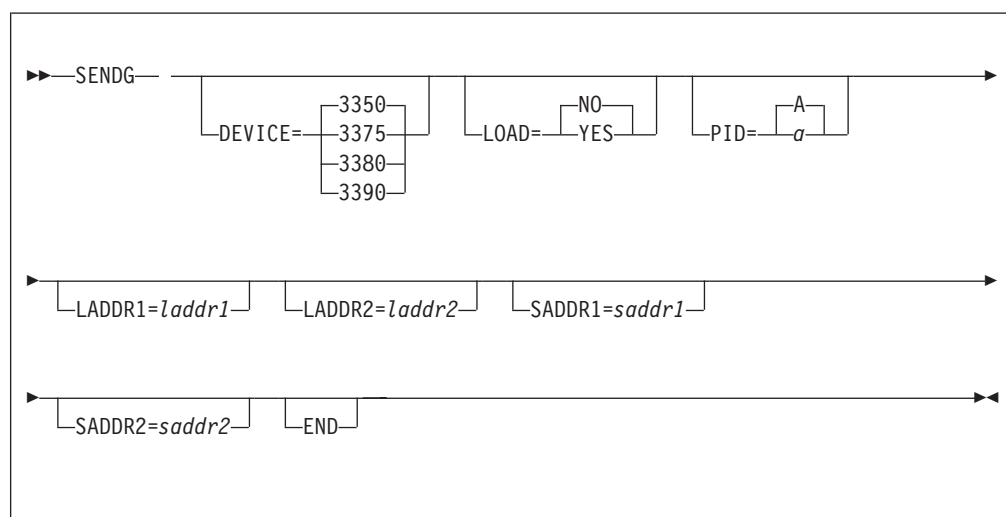
SENDG is coded as the last macro in SKLNG/SENDG macro pairs (that represent SCK macro sets), and as the final delimiter at the end of the SCK generation.

The very first SENDG macro coded is coded with the DEVICE, LOAD, and PID options. If the first SENDG macro does not contain any STC options, then the default selections are set.

Subsequent SENDG macro calls are coded without parameters.

The final SENDG macro (the delimiter) is coded with the END option, and is coded following the last PKSTG. This invocation performs final checks, generates PDUMP records, if required, and provides a report of the number of records of each type that were generated.

## Format



### **DEVICE=3350|3375|3380|3390**

This parameter describes the disk device used for the SDMF/DRIL disk. 3390s installed in 3380 emulation mode must be coded as 3380.

### **LOAD=YES|NO**

Yes is used when the customer has changed the format of the SCK or PKST records and wishes to load the SDMF and DRIL disk with updated record DSECTs.

### **PID=A|a**

This parameter provides a single alphabetic character to identify the SCK pilot tape from other pilot tapes. This parameter must be alphabetic and if omitted the alphabetic character 'A' will be substituted.

### **LADDR1=laddr1**

This parameter specifies the starting address of large records on the pack.

### **LADDR2=laddr2**

This parameter specifies the ending address of large records on the pack.

## SENDG

**SADDR1**=*saddr1*

This parameter specifies the starting address of small records on the pack.

**SADDR2**=*saddr2*

This parameter specifies the ending address of small records on the pack.

**END**

The END parameter is only coded on the last SENDG macro statement in the input stream.

**Note:** The LADDR1, LADDR2, SADDR1, and SADDR2 parameters are used to override the defaults on the STC load address control cards. If these parameters are not coded, the defaults defined in SENDG (see listing) are used.

## Examples

None.

## References

None.



## Creating the Non-SNA Communications Pilot Tape

Output from Phase I can be on cards or tape depending upon the data definition of SYSPUNCH in the Phase I JCL. This is used as input, CARDIN, to STC to produce a non-SNA communications pilot tape, TAPE1.

For instructions for running Phase II see “System Test Compiler” in *TPF Operations*.

The following STC control cards are generated in Phase I.

If DEVICE=3350 is coded on the SENDG macro, the following Load Address statements will be generated:

```
LOAD ADDRESSES 0A01 TO 2E0B,N,1055 1
LOAD ADDRESSES 2F00 TO 690B,N,381
```

**Note:** If DEVICE=3375, 3380, or 3390, the load address statements generated are the same as for DEVICE=3350.

In addition to the above, the following STC Control Statements will also be generated:

```
STC LOAD DISKFILE    {3350} { }
                    {3375}
                    {3380}
                    {3390}
RUNID PILOT {A}      PILOT
                  {a}
DATA
```

where braces { } indicate alternative values.

If these disk allocation control cards do not correspond to the type or format of the users unit test disk (SDMF and DRIL), they must be changed before running Phase II, or the GENSTC macro must be changed prior to Phase I, to produce the proper cards. This is especially important if the user's SDMF and DRIL files are not located at the place specified in these control cards. For an explanation of the information on the control cards, refer to information about the system test compiler in *TPF Operations*.

## Inner Macros

Macro Name:	<b>CVHDK</b>
Prototype:	&NAME CVHDK &CHARS
Function:	This macro validates hexadecimal input characters and converts them to decimal. From one to seven digits is allowable within the range of 1–FFFFFFF.
Restrictions:	No more than seven digits will be converted.
Input:	One positional parameter, the hexadecimal value to be converted to decimal, is the only input.
Output:	The results of the conversion are stored in the global SETA variable &XADEC. If &XADEC is zero after CVHDK has been called invalid, hexadecimal characters have been submitted for conversion and an error message must be issued.
Diagnostics:	&CHARS
	Must be one to seven digits, from X'1–FFFFFFF'. Each digit must be from 0–F.
Logic Aids:	None
Dependencies:	None
Macros Used:	None
Error Messages:	None

## SENDG

Calling Macros: GENSTC, ISKLG, PKSTG, SKLNG

Macro Name: **CVRTK**  
Prototype: &NAME CVRTK &P1  
Function: This macro converts a decimal number into a hexadecimal number.  
Restrictions: None  
Input: One positional parameter, the decimal value to be converted to hexadecimal, is the only input.  
Output: The results of the conversion are stored in the global SETA variable &STCDEC. If &STCDEC is zero after CVRTK has been called, the value submitted for conversion was not numeric and an error message must be issued.  
Diagnostics: &P1 must be numeric.  
Logic Aids: None  
Dependencies: None  
Macros Used: None  
Error Messages: None  
Calling Macros: GENSTC, ISKLG, PKSTG, RLSTG, SENDG, SKLNG

Macro Name: **GENSTC**  
Prototype: &NAME GENSTC  
Function: This macro is called by SENDG to generate the STC input for creating an SCK record.  
Restrictions: None.  
Input: No direct input is provided. All global output from the SCK generation user and inner macros is available to GENSTC.  
Output: STC input to create the SCK records. This output is normally placed on tape. The first GENSTC call will generate STC control cards for creating the non-SNA communications pilot tape.  
Diagnostics: None  
Logic Aids: None  
Dependencies: None  
Macros Used: CVHDK, CVRTK, SETX  
Error Messages: None  
Calling Macros: SENDG

Macro Name: **ISKLG**  
Prototype: &NAME ISKLG  
Function: This macro is called by SENDG to generate the assembly listing for that portion of the SCK which is common to both local and remote lines. It performs checks and sets values for use by GENSTC in generating the STC input which creates the common area of the SCK.  
Restrictions: None  
Input: No direct input is provided. All global output from the SCK generation user macros is available to ISKLG, but for the most part this macro handles the output from the user macro SKLNG.  
Output: An assembly listing of the SCK common area. Global fields are set up for use by GENSTC in generating the STC input. Object code is generated by ISKLG and output to the same device as the STC input generated by GENSTC. However, STC will ignore the object code when it reads the input data.  
Diagnostics: None  
Logic Aids: None  
Dependencies: GENSTC  
Macros Used: CVHDK, CVRTK, SETX

Error Messages:	<p>The error messages issued by this macro refer to errors which were discovered during execution of the SKLNG macro. They are put out by ISKLG so that they will appear in the assembly listing as close as possible to the field which will need correction.</p> <p>4, INVALID LINE IN PATH PARAMETER—DEFAULTED TO VOID PATH</p> <p>4, INVALID CU IN PATH PARAMETER—DEFAULTED TO VOID PATH</p> <p>4, INVALID PATH INDICATOR—MA ASSUMED</p> <p>4, NO PRIME PATH—DEFAULTED TO NONVOID</p> <p>4, NUMBER OF PATHS SPECIFIED GT MAX SPECIFIED IN SIP</p> <p>4, INVALID BSC SEND LEVEL. DEFAULT = 1</p> <p>4, NUMBER OF TIS EXCEEDS MAX—DEFAULTED TO MAXTI</p>
Calling Macros:	SENDG
Macro Name:	<b>SETX</b>
Prototype:	&NAME SETX &TI,&OP,&T2
Function:	This macro validates two or more hexadecimal numbers and converts them to decimal; it performs the indicated operation and then converts the result into hexadecimal.
Restrictions:	Each hexadecimal number specified may be no more than seven digits in the range 1–FFFFFFF.
Input:	<p>Three positional parameters:</p> <p>&amp;TI</p> <p>the operand, a hexadecimal number in the range, 1–FFFFFFF.</p> <p>&amp;OP</p> <p>the operation,</p> <p>A - Add</p> <p>S - Subtract</p> <p>D - Divide</p> <p>M - Multiply</p> <p>&amp;T2</p> <p>the operator, for subtraction, division and multiplication this is a single hex number in the range 1–FFFFFFF. For addition it may be a list of up to 15 subparameters, each one a hexadecimal number in the range 1–FFFFFFF.</p>
Output:	The results of the computation are stored in the global SETA variable &STCDEC. If &STCDEC is zero after SETX has been called an error has occurred and an error message must be issued.
Diagnostics:	<p>&amp;TI</p> <p>must be one to seven hexadecimal digits in the range 1–FFFFFFF.</p> <p>if invalid, it will be treated as zero.</p> <p>&amp;OP</p> <p>must be A, S, D or M.</p> <p>if invalid, it will be treated as M.</p> <p>&amp;T2</p> <p>for addition, may be a list of up to 15 subparameters.</p> <p>if more than one subparameter is entered for subtraction, multiplication, or division, only the first will be used.</p> <p>each subparameter must be one to seven hexadecimal digits in the range 1–FFFFFFF.</p>
Logic Aids:	None.
Dependencies:	None.
Macros Used:	CVHDK, CVRTK
Error Messages:	None.
Calling Macros:	GENSTC, ISKLG

**SENDG**

---

## Appendix A. System Generation Messages

|                      The system generation error messages have been moved to *Messages (System*  
|                      *Error and Offline)*.



## Appendix B. Sample SIP Input and Report

This chapter consists of a sample SIP Stage I input deck. The input deck defines a single-module, fully duplicated, HPO, MDBF, and non-RES SNA-supported system.

### Sample SIP Stage I Input

**Note:** This example is meant to show the method of coding the SIP user macros. The values coded do *not* represent any recommended values *nor* does this represent a usable system. Not everything defined in this system is required to operate a TPF system. Definitions are required for customer applications.

The user should keep in mind the following when coding the JCL:

1. SYSLIB data definition identifies the location of SIP Stage I macros and the SYFCTB macro created by the FCTBG.
2. SYSPUNCH data definition is related to the output of Stage I which is input to Stage II. In this example, a tape is shown and recommended. The tape has a standard label of STAGE1. As an alternative, the output of Stage I may be a card deck with the data definition stating SYSOUT=B. In turn, this card deck would be input to Stage II.
3. The SIP Stage I macros follow the SYSIN card. They have no specific sequence except that the GENSIP macro must be the last macro that appears before the assembler END card.
4. If the SYSPUNCH card refers to a tape, this tape may need to be initialized, while the SYSPUNCH card and Start command for Stage II require modification. The important point is that the tape initialization, data definition, and the Start command must agree. See the IEHINITT program in *OS/VS2 MVS Utilities*.

```
//SIPSTG1      JOB  SIPGEN,USER,CLASS=A,REGION=400K
//ASMCP      EXEC  PGM=ASMA90,REGION=3072K,TIME=120,PARM='DECK,NOLoad'
//SYSLIB      DD   DSN=ACP.SYMACRO.RELxx,DISP=SHR
//           DD   DSN=ACP.SIPGEN.RELxx,DISP=SHR
//           DD   DSN=ACP.MACRO.RELxx,DISP=SHR
//SYSUT1      DD   UNIT=SYSDA,SPACE=(CYL,(20,10,20))
//SYSUT2      DD   UNIT=SYSDA,SPACE=(CYL,(20,10,20))
//SYSUT3      DD   UNIT=SYSDA,SPACE=(CYL,(20,10,20))
//SYSGO      DD   DUMMY
//SYSPUNCH    DD   DSN=STAGE1,UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(,SL)
//SYSPRINT    DD   SYSOUT=A,DCB=(BLKSIZE=133,RECFM=FBM,LRECL=121),
//           SPACE=(TRK,(100,200),RLSE),OUTLIM=500000
//SYSIN       DD   *
*

      TITLE 'TPF 4.1 BASIC SUBSYSTEM - TCP'
      PRINT NOGEN
*****
*
*      TPF 4.1 SYSTEM TEST TCP BASIC SUBSYSTEM
*
*      FULLY DUPLICATED
*      2 3390 * 1113 CYL device A
*      2 3390 * 1113 CYL device B
*****
      EJECT 1
*
*
*
*****
```

```

*                SSDEF - SUBSYSTEM USER DEFINITION                *
*****
*
    SSDEF      TOTSS=4,          TOTAL NUMBER OF SUBSYSTEMS      X
               TOTSSU=8,        MAX NUMBER OF SSUSERS IN COMPLEX X
               SSNAME=BSS,      BASIC SUBSYSTEM                  X
               BSSGEN=YES,      BASIC SUBSYSTEM                  X
               SSUID1=(HPN,HPN2) TWO SSUSERS
    EJECT 1
*
*
*
*****
*                CONFIG - SYSTEM HARDWARE SUPPORT                  *
*****
*
    CONFIG     APRNT=00E,        ADDRESS OF PRINTER FOR IPL SEQUENCE X
               DUMPDEV=PRTR,     UPON IPL DUMPS ARE SENT TO PRINTER X
               FQTK=NO,          NO FARE QUOTE TICKETING          X
               MAPSP=YES,        INCLUDE 3270 MAPPING SUPPORT      X
               MSGSW=YES,        INCLUDE MESSAGE SWITCHING SUPPORT X
               USER=USA,         USER OF PARS                     X
               DCUSV=16,         ADDR. RANGE DASD CONTROL RANGE   X
               ENTERPRISE=DANBURY,
               COMPLEX=TPFNET,    X
               SYSID=(B,C,D,E,Z,0,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,XX
               ,Y,1,2,3,4,5,6),   X
               PROC1=(20410,3090,,1), TPFXA1 LPAR                X
               PROC2=(30410,3090,0,,0), TPF1 LPAR                 X
               PROC3=(40410,3090,,2,,3), TPF2 LPAR (ALSO 3090A/3090B) X
               PROC4=(50410,3090,4,,4), TPF3 LPAR                 X
               PROC5=(D0105,9221), OLYMP processor               X
               PROC6=(20215,9121), KGN test floor processor       X
               PROC7=(20023,9021), POK test floor processor       X
               PROC8=(20024,9021), Our 8th Processor              X
               PROC9=(20025,9021), Our 9th Processor              X
               PROC10=(20026,9021), Our 10th processor            X
               PROC11=(20027,9021), Our 11th processor            X
               PROC12=(20028,9021), Our 12th processor            X
               PROC13=(20029,9021), Our 13th processor            X
               PROC14=(20030,9021), Our 14th processor            X
               PROC15=(20031,9021), Our 15th processor            X
               PROC16=(20032,9021), Our 16th processor            X
               PROC17=(20033,9021), Our 17th processor            X
               PROC18=(20034,9021), Our 18th processor            X
               PROC19=(20035,9021), Our 19th processor            X
               PROC20=(20036,9021), Our 20th processor            X
               PROC21=(20037,9021), Our 21th processor            X
               PROC22=(20038,9021), Our 22th processor            X
               PROC23=(20039,9021), Our 23th processor            X
               PROC24=(20040,9021), Our 24th processor            X
               PROC25=(20041,9021), Our 25th processor            X
               PROC26=(20042,9021), Our 26th processor            X
               PROC27=(20043,9021), Our 27th processor            X
               PROC28=(20044,9021), Our 28th processor            X
               PROC29=(20045,9021), Our 29th processor            X
               PROC30=(20046,9021), Our 30th processor            X
               PROC31=(20047,9021), Our 31th processor            X
               PROC32=(20048,9021), Our 32th processor            X
               VM=YES,           ACP WILL RUN UNDER VM/370        X
               RES=NO, DONT      INCLUDE RES APPLICATION          X
               TEST=NO,          WP TEST SYSTEM PARMETER          X
               ACF=YES,          ACF FEATURE                      X
               MPIF=YES,         MPIF FEATURE                     X
               CIPHR=(BOTH,ACP.OBJ.OC040), BOTH ENCRYPTION TECHNIQUES X
               WTOPCUNS=YES,
               VEQR=NO,          VIRTUAL = VIRTUAL                X

```



SELACT=YES,	Enable E-type Loader Sel. activate	X
TPPDF=YES,	TPPDF INCLUDED IN SYSTEM	X
TPFAR=YES,		X
CTKI32LC=YES,	GENERATE CTKI IN 32LC FORMAT	X
NEF=YES		@000.000

# EJECT 1

```

*****
*****
*          START IODEV - DEFINES VALID DEVICE ADDRESS          *
*****
*
*****
*          ***** DASD *****          *
*****
IODEV      IOADR=0110,DVTYP=DASD          KGN
IODEV      IOADR=0118,DVTYP=DASD          POK 710
IODEV      IOADR=0120,DVTYP=DASD          KGN
IODEV      IOADR=0210,DVTYP=DASD          KGN
IODEV      IOADR=0220,DVTYP=DASD          KGN
IODEV      IOADR=0260,DVTYP=DASD          VPARS
IODEV      IOADR=03E0,DVTYP=DASD          VPARS
IODEV      IOADR=0460,DVTYP=DASD          VPARS
IODEV      IOADR=0490,DVTYP=DASD          POK @000.000
IODEV      IOADR=04A0,DVTYP=DASD          POK @000.000
IODEV      IOADR=04E0,DVTYP=DASD          VM TESTING
IODEV      IOADR=04F0,DVTYP=DASD          VM TESTING
IODEV      IOADR=0520,DVTYP=DASD          VM TESTING
IODEV      IOADR=0760,DVTYP=DASD          POK
IODEV      IOADR=0768,DVTYP=DASD          POK
IODEV      IOADR=0A10,DVTYP=DASD          KGN
IODEV      IOADR=0A20,DVTYP=DASD          KGN
IODEV      IOADR=0A40,DVTYP=DASD          POK
IODEV      IOADR=0A48,DVTYP=DASD          POK
IODEV      IOADR=0A60,DVTYP=DASD          POK
IODEV      IOADR=0A68,DVTYP=DASD          POK
IODEV      IOADR=0C40,DVTYP=DASD          POK
IODEV      IOADR=0C48,DVTYP=DASD          POK
IODEV      IOADR=0DC0,DVTYP=DASD          POK
IODEV      IOADR=0DC8,DVTYP=DASD          POK
IODEV      IOADR=0DE0,DVTYP=DASD          POK
IODEV      IOADR=0DE8,DVTYP=DASD          POK
IODEV      IOADR=0E30,DVTYP=DASD          MVSLBx
IODEV      IOADR=0E38,DVTYP=DASD          MVSLBx
IODEV      IOADR=0E80,DVTYP=DASD          VPARS
IODEV      IOADR=0EE0,DVTYP=DASD          VPARS
IODEV      IOADR=0EE8,DVTYP=DASD          VPARS
IODEV      IOADR=1440,DVTYP=DASD          POK 710
IODEV      IOADR=1448,DVTYP=DASD          POK 710
IODEV      IOADR=1E10,DVTYP=DASD          SUTTER
IODEV      IOADR=1E40,DVTYP=DASD,RCSSID=0003  RUSSELL
IODEV      IOADR=1EC0,DVTYP=DASD,RCSSID=0008  BODMER @000.000
IODEV      IOADR=1EC8,DVTYP=DASD,RCSSID=0008  BODMER @000.000
IODEV      IOADR=1ED0,DVTYP=DASD,RCSSID=0008  BODMER @000.000
IODEV      IOADR=1ED8,DVTYP=DASD,RCSSID=0008  BODMER @000.000
IODEV      IOADR=1400,DVTYP=DASD          Chloe @000.000
IODEV      IOADR=1408,DVTYP=DASD          Chloe @000.000
IODEV      IOADR=1410,DVTYP=DASD          Chloe @000.000
IODEV      IOADR=1418,DVTYP=DASD          Chloe @000.000
IODEV      IOADR=1120,DVTYP=DASD          SAs/Harvey
IODEV      IOADR=1140,DVTYP=DASD          URFER
IODEV      IOADR=1148,DVTYP=DASD          URFER
IODEV      IOADR=1150,DVTYP=DASD          URFER
IODEV      IOADR=1158,DVTYP=DASD          URFER
IODEV      IOADR=1160,DVTYP=DASD          URFER
IODEV      IOADR=1168,DVTYP=DASD          URFER
IODEV      IOADR=1170,DVTYP=DASD          URFER
IODEV      IOADR=1178,DVTYP=DASD          URFER

```

IODEV	IOADR=1180,DVTYP=DASD	KRUGER
IODEV	IOADR=1188,DVTYP=DASD	KRUGER
IODEV	IOADR=1190,DVTYP=DASD	KRUGER
IODEV	IOADR=1198,DVTYP=DASD	KRUGER
IODEV	IOADR=11A0,DVTYP=DASD	KRUGER
IODEV	IOADR=11A8,DVTYP=DASD	KRUGER
IODEV	IOADR=11B0,DVTYP=DASD	KRUGER
IODEV	IOADR=11B8,DVTYP=DASD	KRUGER
IODEV	IOADR=11C0,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11C8,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11D0,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11D8,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11E0,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11E8,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11F0,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=11F8,DVTYP=DASD	E-ticket @000.000
IODEV	IOADR=0D00,DVTYP=DASD	Scully @000.000
IODEV	IOADR=0D08,DVTYP=DASD	Scully @000.000
IODEV	IOADR=0D10,DVTYP=DASD	Scully @000.000
IODEV	IOADR=0D18,DVTYP=DASD	Scully @000.000
IODEV	IOADR=2000,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2008,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2010,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2018,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2020,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2028,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2030,DVTYP=DASD	Shark @000.000
IODEV	IOADR=2038,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7000,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7008,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7010,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7018,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7040,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7048,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7050,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7058,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7080,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7088,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7090,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7098,DVTYP=DASD	Shark @000.000
IODEV	IOADR=70C0,DVTYP=DASD	Shark @000.000
IODEV	IOADR=70C8,DVTYP=DASD	Shark @000.000
IODEV	IOADR=70D0,DVTYP=DASD	Shark @000.000
IODEV	IOADR=70D8,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7100,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7108,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7110,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7118,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7140,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7148,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7150,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7158,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7180,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7188,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7190,DVTYP=DASD	Shark @000.000
IODEV	IOADR=7198,DVTYP=DASD	Shark @000.000
IODEV	IOADR=71C0,DVTYP=DASD	Shark @000.000
IODEV	IOADR=71C8,DVTYP=DASD	Shark @000.000
IODEV	IOADR=71D0,DVTYP=DASD	Shark @000.000
IODEV	IOADR=71D8,DVTYP=DASD	Shark @000.000
IODEV	IOADR=0D40,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D48,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D50,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D58,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D60,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D68,DVTYP=DASD	O.E.M. @000.000
IODEV	IOADR=0D80,DVTYP=DASD	O.E.M. @000.000

IODEV	IOADR=0D88,DVTYP=DASD	O.E.M.	@000.000
IODEV	IOADR=0D90,DVTYP=DASD	O.E.M.	@000.000
IODEV	IOADR=0D98,DVTYP=DASD	O.E.M.	@000.000
IODEV	IOADR=0DA0,DVTYP=DASD	O.E.M.	@000.000
IODEV	IOADR=0DA8,DVTYP=DASD	O.E.M.	@000.000
*****			
*	***** TAPE *****		*
*****			
IODEV	IOADR=42,DVTYP=TAPE	VM TESTING	
IODEV	IOADR=43,DVTYP=TAPE	VM TESTING	
IODEV	IOADR=44,DVTYP=TAPE	VTS	@000.000
IODEV	IOADR=45,DVTYP=TAPE	VTS	@000.000
* IODEV	IOADR=46,DVTYP=TAPE	VTS	@000.000
IODEV	IOADR=47,DVTYP=TAPE	VTS	@000.000
* IODEV	IOADR=4C,DVTYP=TAPE	VM VTS Testing	
IODEV	IOADR=4D,DVTYP=TAPE	VM VTS Testing	
IODEV	IOADR=57,DVTYP=TAPE	VM TESTING	
IODEV	IOADR=58,DVTYP=TAPE	VM TESTING	
IODEV	IOADR=5D,DVTYP=TAPE	XA1 TPF LPARS	
IODEV	IOADR=77,DVTYP=TAPE	POK/KGN	
IODEV	IOADR=F1,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F2,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F3,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F4,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F5,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F6,DVTYP=TAPE	XA1/TPF LPARS	
IODEV	IOADR=F8,DVTYP=TAPE	3590S	
IODEV	IOADR=FC,DVTYP=TAPE	VTS-3590s	@000.000
IODEV	IOADR=FD,DVTYP=TAPE	VTS-3590s	@000.000
IODEV	IOADR=FE,DVTYP=TAPE	3495S	@000.000
*****			
*	***** 37x5 *****		*
*****			
IODEV	IOADR=00B3,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=00B4,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0340,DVTYP=37X5		
IODEV	IOADR=0701,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0702,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0704,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0705,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0706,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0711,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0712,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0715,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0716,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0718,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=071A,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0721,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0722,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0724,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0741,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0742,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0743,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0744,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0745,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0746,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0747,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0748,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0749,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0861,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0862,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0864,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0865,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0866,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0867,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0868,DVTYP=37X5	XA1/TPF LPARS	
IODEV	IOADR=0869,DVTYP=37X5	XA1/TPF LPARS	

IODEV	IOADR=0875,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=0876,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A2,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A3,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A4,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A5,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A6,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08A7,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08B0,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08B1,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08B2,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08B3,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=08B4,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=0930,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=0940,DVTYP=37X5	XA1/TPF LPARS
IODEV	IOADR=1240,DVTYP=37X5	

\*\*\*\*\*  
 \* \*\*\*\*\* SCTC \*\*\*\*\* \*  
 \*\*\*\*\*

IODEV	IOADR=0394,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=0395,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=0396,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=0397,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=0398,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=0399,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=039A,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=039B,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=039C,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=039E,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03A8,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03A9,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AA,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AB,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AC,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AD,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AE,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03AF,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B0,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B1,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B2,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B3,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B4,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B5,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B6,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B7,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B8,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03B9,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03BA,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03BB,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=03BD,DVTYP=SCTC	XA1
IODEV	IOADR=03BE,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=1390,DVTYP=SCTC	XA1
IODEV	IOADR=1391,DVTYP=SCTC	XA1
IODEV	IOADR=1392,DVTYP=SCTC	XA1
IODEV	IOADR=1393,DVTYP=SCTC	XA1
IODEV	IOADR=139D,DVTYP=SCTC	XA1
IODEV	IOADR=139E,DVTYP=SCTC	XA1
IODEV	IOADR=13A0,DVTYP=SCTC	XA1
IODEV	IOADR=13A1,DVTYP=SCTC	XA1
IODEV	IOADR=13A2,DVTYP=SCTC	XA1
IODEV	IOADR=13A3,DVTYP=SCTC	XA1
IODEV	IOADR=13A8,DVTYP=SCTC	XA1
IODEV	IOADR=13A9,DVTYP=SCTC	XA1
IODEV	IOADR=13AA,DVTYP=SCTC	XA1
IODEV	IOADR=13AB,DVTYP=SCTC	XA1
IODEV	IOADR=13B8,DVTYP=SCTC	XA1
IODEV	IOADR=13B9,DVTYP=SCTC	XA1

IODEV	IOADR=13BA,DVTYP=SCTC	XA1
IODEV	IOADR=13BB,DVTYP=SCTC	XA1
IODEV	IOADR=13BE,DVTYP=SCTC	XA1
IODEV	IOADR=2380,DVTYP=SCTC	XA1
IODEV	IOADR=2381,DVTYP=SCTC	XA1
IODEV	IOADR=2382,DVTYP=SCTC	XA1
IODEV	IOADR=2383,DVTYP=SCTC	XA1
IODEV	IOADR=2394,DVTYP=SCTC	XA1
IODEV	IOADR=2395,DVTYP=SCTC	XA1
IODEV	IOADR=2396,DVTYP=SCTC	XA1
IODEV	IOADR=2397,DVTYP=SCTC	XA1
IODEV	IOADR=2398,DVTYP=SCTC	XA1
IODEV	IOADR=2399,DVTYP=SCTC	XA1
IODEV	IOADR=239A,DVTYP=SCTC	XA1
IODEV	IOADR=239B,DVTYP=SCTC	XA1
IODEV	IOADR=239C,DVTYP=SCTC	XA1
IODEV	IOADR=23A0,DVTYP=SCTC	XA1
IODEV	IOADR=23A1,DVTYP=SCTC	XA1
IODEV	IOADR=23A2,DVTYP=SCTC	XA1
IODEV	IOADR=23A3,DVTYP=SCTC	XA1
IODEV	IOADR=23A4,DVTYP=SCTC	XA1
IODEV	IOADR=23A5,DVTYP=SCTC	XA1
IODEV	IOADR=23A6,DVTYP=SCTC	XA1
IODEV	IOADR=23A7,DVTYP=SCTC	XA1
IODEV	IOADR=23A8,DVTYP=SCTC	XA1
IODEV	IOADR=23A9,DVTYP=SCTC	XA1
IODEV	IOADR=23AA,DVTYP=SCTC	XA1
IODEV	IOADR=23AB,DVTYP=SCTC	XA1
IODEV	IOADR=23AC,DVTYP=SCTC	XA1
IODEV	IOADR=23AD,DVTYP=SCTC	XA1
IODEV	IOADR=23AE,DVTYP=SCTC	XA1
IODEV	IOADR=23AF,DVTYP=SCTC	XA1
IODEV	IOADR=23B4,DVTYP=SCTC	XA1
IODEV	IOADR=23B5,DVTYP=SCTC	XA1
IODEV	IOADR=23B6,DVTYP=SCTC	XA1
IODEV	IOADR=23B7,DVTYP=SCTC	XA1
IODEV	IOADR=23B8,DVTYP=SCTC	XA1
IODEV	IOADR=23B9,DVTYP=SCTC	XA1
IODEV	IOADR=23BA,DVTYP=SCTC	XA1
IODEV	IOADR=23BB,DVTYP=SCTC	XA1
IODEV	IOADR=23BC,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=23BD,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=23BE,DVTYP=SCTC	XA1/MVSESA
IODEV	IOADR=3380,DVTYP=SCTC	XA1
IODEV	IOADR=3381,DVTYP=SCTC	XA1
IODEV	IOADR=3382,DVTYP=SCTC	XA1
IODEV	IOADR=3383,DVTYP=SCTC	XA1
IODEV	IOADR=3390,DVTYP=SCTC	XA1
IODEV	IOADR=3391,DVTYP=SCTC	XA1
IODEV	IOADR=3392,DVTYP=SCTC	XA1
IODEV	IOADR=3393,DVTYP=SCTC	XA1
IODEV	IOADR=3394,DVTYP=SCTC	XA1
IODEV	IOADR=3395,DVTYP=SCTC	XA1
IODEV	IOADR=3396,DVTYP=SCTC	XA1
IODEV	IOADR=3397,DVTYP=SCTC	XA1
IODEV	IOADR=3398,DVTYP=SCTC	XA1
IODEV	IOADR=3399,DVTYP=SCTC	XA1
IODEV	IOADR=339A,DVTYP=SCTC	XA1
IODEV	IOADR=339B,DVTYP=SCTC	XA1
IODEV	IOADR=339C,DVTYP=SCTC	XA1
IODEV	IOADR=33A0,DVTYP=SCTC	TPF1
IODEV	IOADR=33A1,DVTYP=SCTC	TPF1
IODEV	IOADR=33A2,DVTYP=SCTC	TPF1
IODEV	IOADR=33A3,DVTYP=SCTC	TPF1
IODEV	IOADR=33AA,DVTYP=SCTC	TPF1
IODEV	IOADR=33AB,DVTYP=SCTC	TPF1
IODEV	IOADR=33B0,DVTYP=SCTC	TPF1

IODEV	IOADR=33B1, DVTYP=SCTC	TPF1
IODEV	IOADR=33B2, DVTYP=SCTC	TPF1
IODEV	IOADR=33B3, DVTYP=SCTC	TPF1
IODEV	IOADR=33B8, DVTYP=SCTC	TPF1
IODEV	IOADR=33B9, DVTYP=SCTC	TPF1
IODEV	IOADR=33BA, DVTYP=SCTC	TPF1
IODEV	IOADR=33BB, DVTYP=SCTC	TPF1
IODEV	IOADR=33BC, DVTYP=SCTC	TPF1
IODEV	IOADR=33BD, DVTYP=SCTC	TPF1
IODEV	IOADR=33BE, DVTYP=SCTC	TPF1
IODEV	IOADR=43A4, DVTYP=SCTC	TPF2
IODEV	IOADR=43A5, DVTYP=SCTC	TPF2
IODEV	IOADR=43A6, DVTYP=SCTC	TPF2
IODEV	IOADR=43A7, DVTYP=SCTC	TPF2
IODEV	IOADR=43AA, DVTYP=SCTC	TPF2
IODEV	IOADR=43AB, DVTYP=SCTC	TPF2
IODEV	IOADR=43AC, DVTYP=SCTC	TPF2
IODEV	IOADR=43AD, DVTYP=SCTC	TPF2
IODEV	IOADR=43AE, DVTYP=SCTC	TPF2
IODEV	IOADR=43AF, DVTYP=SCTC	TPF2
IODEV	IOADR=43B4, DVTYP=SCTC	TPF2
IODEV	IOADR=43B5, DVTYP=SCTC	TPF2
IODEV	IOADR=43B6, DVTYP=SCTC	TPF2
IODEV	IOADR=43B7, DVTYP=SCTC	TPF2
IODEV	IOADR=43BC, DVTYP=SCTC	TPF2
IODEV	IOADR=43BD, DVTYP=SCTC	TPF2
IODEV	IOADR=43BE, DVTYP=SCTC	TPF2
IODEV	IOADR=53A8, DVTYP=SCTC	TPF3
IODEV	IOADR=53A9, DVTYP=SCTC	TPF3
IODEV	IOADR=53AA, DVTYP=SCTC	TPF3
IODEV	IOADR=53AB, DVTYP=SCTC	TPF3
IODEV	IOADR=53B0, DVTYP=SCTC	TPF3
IODEV	IOADR=53B1, DVTYP=SCTC	TPF3
IODEV	IOADR=53B2, DVTYP=SCTC	TPF3
IODEV	IOADR=53B3, DVTYP=SCTC	TPF3
IODEV	IOADR=53B8, DVTYP=SCTC	TPF3
IODEV	IOADR=53B9, DVTYP=SCTC	TPF3
IODEV	IOADR=53BA, DVTYP=SCTC	TPF3
IODEV	IOADR=53BB, DVTYP=SCTC	TPF3
IODEV	IOADR=53BC, DVTYP=SCTC	TPF3
IODEV	IOADR=53BD, DVTYP=SCTC	TPF3
IODEV	IOADR=53BE, DVTYP=SCTC	TPF3

```
*****
*           END IODEV - DEFINES VALID DEVICE ADDRESS           *
*****
```

EJECT 1

```
*****
*          CRASTB - PRIME AND RO      (NATIVE CONSOLE SUPPORT)      *
*****
```

```
*  
CRASTB      PRCRS=(22,23,23,60,61,62,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,X  
             1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1,A1),                X  
            RO CRS=(24,24,27,64,64,67,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,X  
             2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2,A2),              X  
            NCONSL=YES,                                                        X  
            CTKC32LC=YES               GENERATE CTKC IN 32LC FORMAT
```

\* SPACE 5

\*\*\*\*\*

```
*          CRAFTB - PRIME AND RO  (NONNATIVE CONSOLE SUPPORT)      *
```

EJECT 1

```
*      CRASTB    PRCRS=(56,56,1F,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56),
*                  6,56,56,56,56,56,56,56,56,56,56,56,56,56,56,56),
*                  ROCRS=(010002,1052,B,
*                          010002,1052,C,
```



```

*****
*                               UNITRD - UNIT RECORD TYPES AND ADDRESSES                               *
*****
*
UNITRD    UNIT=1403,ADDR=00E,SYMNO=1
UNITRD    UNIT=3211,ADDR=01E,SYMNO=2
UNITRD    UNIT=3211,ADDR=02E,SYMNO=3
UNITRD    UNIT=1403,ADDR=03E,SYMNO=4
UNITRD    UNIT=1403,ADDR=04E,SYMNO=5
UNITRD    UNIT=3505,ADDR=00C,SYMNO=1
UNITRD    UNIT=3505,ADDR=00F,SYMNO=2
UNITRD    UNIT=2540R,ADDR=01F,SYMNO=3
*
*****
*                               GLOBAL - CORE RESERVED FOR KEYPOINTS                               *
*****
*
GLOBAL    SSUI SHR1=(2,2),SSUI SHR2=(3,3),SSUI SHR3=(4,4),                                X
          SSUI UNQ1=(4,4),SSUI UNQ2=(4,4),SSUI UNQ3=(4,4),                                X
          SSUX SHR1=(2,2),SSUX SHR2=(3,3),SSUX SHR3=(2,2),                                X
          SSUX UNQ1=(2,2),SSUX UNQ2=(3,3),SSUX UNQ3=(2,2),                                X
          SSULMOD=(7,7),,KEYUPD=60,                                                        X
          AMSGRC=YES,                               INCLUDE APPLICATION MSG RECOVERY        X
          ARPKP=YES,                               INCLUDE APPLICATION RECOVERY KEYPT      X
          RCYTO=5,                                APPLIC MSG RECOVERY TIMEOUT IN SECS
*
*
*****
*                               GLSYNC
*****
*
*                               I-STREAM UNIQUE / SSU UNIQUE RECORDS
*
GLSYNC    GLOBAL=@TSTR0C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=NO,USER=HPN,  X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR0C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=NO,USER=HPN2, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR1C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN,  X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR1C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN2, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR2C,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=NO,USER=HPN,  X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR2C,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=NO,USER=HPN2, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR3C,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN,  X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR3C,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN2, X
          ISTREAM=UNIQUE
*
*                               I-STREAM UNIQUE / SSU COMMON RECORDS
*
GLSYNC    GLOBAL=@TSTR4C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=YES,USER=HPN, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR4C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=YES,USER=HPN2,X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR5C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR5C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN2,X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR6C,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=YES,USER=HPN, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR6C,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=YES,USER=HPN2,X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR7C,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN, X
          ISTREAM=UNIQUE
GLSYNC    GLOBAL=@TSTR7C,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN2,X

```



```

        ISTREAM=UNIQUE
GLSYNC GLOBAL=@ZCREC,BASE=Y,TYPE=FLD,PROTECT=3,COMMON=YES,USER=HPN, X
        ISTREAM=UNIQUE,LOAD=YES
*
*      I-STREAM SHARED / SSU UNIQUE RECORDS
*
GLSYNC GLOBAL=@TSTR8C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=NO,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTR8C,BASE=A,TYPE=RCD,PROTECT=1,COMMON=NO,USER=HPN2, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTR9C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTR9C,BASE=A,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN2, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRAC,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=NO,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRAC,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=NO,USER=HPN2, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRBC,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRBC,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=NO,USER=HPN2, X
        ISTREAM=SHARED
*
*      I-STREAM SHARED / SSU COMMON RECORDS
*
GLSYNC GLOBAL=@TSTRCC,BASE=A,TYPE=RCD,PROTECT=1,COMMON=YES,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRCC,BASE=A,TYPE=RCD,PROTECT=1,COMMON=YES,USER=HPN2,X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRDC,BASE=A,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRDC,BASE=A,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN2,X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTREC,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=YES,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTREC,BASE=Y,TYPE=RCD,PROTECT=3,COMMON=YES,USER=HPN2,X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRFC,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN, X
        ISTREAM=SHARED
GLSYNC GLOBAL=@TSTRFC,BASE=Y,TYPE=RCD,PROTECT=2,COMMON=YES,USER=HPN2,X
        ISTREAM=SHARED
*
*      I-STREAM UNIQUE / SSU UNIQUE INDICATOR FIELD
*
GLSYNC GLOBAL=@TSTFLD0,BASE=A,TYPE=FLD,PROTECT=1,USER=HPN, X
        COMMON=NO,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD0,BASE=A,TYPE=FLD,PROTECT=1,USER=HPN2, X
        COMMON=NO,LOAD=YES,ISTREAM=UNIQUE
*
*      I-STREAM UNIQUE FIELDS
*
GLSYNC GLOBAL=@TSTFLD1,BASE=Y,TYPE=FLD,PROTECT=3,USER=HPN, X
        COMMON=YES,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD1,BASE=Y,TYPE=FLD,PROTECT=3,USER=HPN2, X
        COMMON=YES,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD2,BASE=A,TYPE=FLD,PROTECT=1,USER=HPN, X
        COMMON=NO,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD2,BASE=A,TYPE=FLD,PROTECT=1,USER=HPN2, X
        COMMON=NO,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD3,BASE=Y,TYPE=FLD,PROTECT=3,USER=HPN, X
        COMMON=YES,LOAD=YES,ISTREAM=UNIQUE
GLSYNC GLOBAL=@TSTFLD3,BASE=Y,TYPE=FLD,PROTECT=3,USER=HPN2, X
        COMMON=YES,LOAD=YES,ISTREAM=UNIQUE
*
*
*****
*      RAM - RESOURCE ALLOCATIONS AND LIST PARAMETERS      *

```

```

*****
*
*      RAM      ECBNL=NOLIMIT,      NBR OF NESTING LEVELS FOR EACH ECB  X
*               GFENS=60,           NUMBER OF GENERAL FILE SLOTS      X
*               GFMOD=010,          SYMBOLIC MOD NBR - 1ST GENERAL FILE X
*               GSON=59,            MAX NBR OF SON GF STATUS TBL SLOTS  X
*               NFBACK=0,           NUMBER OF KEYPOINT FALLBACK AREAS   X
*               PHTBL=257,          NBR OF PRIMARY HOLD TABLE ENTRIES  X
*               OHTBL=1001,         NBR OF OVERFLOW HOLD TABLE ENTRIES X
*               OLDXPAT=1000,       XTRA PAT SLOTS                      X
*               HASHSZ=0           NUM OF BYTES FOR HASH TABLE
*
*      EJECT 1
*
*
*
*****
*      ONLFIL - ONLINE DEVICE CHARACTERISTICS      *
*****
*
*      ONLFIL  DEVICEA=(3390,3339), TYPE OF DEVICE      X
*              DUPTYP A=F,          DUPLICATION STATUS OF DEVICE      X
*              PERMA=4,             NBR OF PERMANENTLY MOUNTED MODULES X
*              IPLABLE=2,           NBR OF IPLABLE MODULES            X
*              VSNCHAR=SN,          ALPHA PORTION OF VSN              X
*              NAMDEVA=DEVA,        X
*              VOLNOA=1,            STARTING VOLUME SERIAL NUMBER      X
*              DEVICEB=(3390,3339), TYPE OF DEVICE      X
*              DUPTYPB=F,          DUPLICATION STATUS OF DEVICE      X
*              PERMB=4,             NBR OF PERMANENTLY MOUNTED MODULES X
*              NAMDEV B=DEV B,      X
*              VOLNOB=5             STARTING VOLUME SERIAL NUMBER
*
*      EJECT 1
*
*
*
*****
*      GENFIL - GENERAL FILE AND DATA SET SUPPORT  *
*****
*
*      GENFIL  DS=2, GSIZE=L, CYL=80, TRK=0, DEV=3390    SON RECOUP
*      GENFIL  DS=3, GSIZE=L, CYL=75, TRK=0, DEV=3390    SON POOL GEN
*      GENFIL  DS=6, GSIZE=4, CYL=0, TRK=1, DEV=3390     4K GF APPLICAT
*
*      EJECT 1
*
*
*
*      EJECT 1                                     @000.000
*****
*      UFTFTI - Establish UFT assignments for FARF4/FARF5 *
*      addressability.                                   *
*****
*      UFTFTI  STAGE=FARF45, MODE=FARF4,                X
*               UFTI=((0,08), (1,10), (2,07), (61,15), (62,13), (63,10), X
*               (51,15), (52,13), (53,10), (10,08), (11,10), (12,07)), X
*               UFTI6=(51,24)                             @000.000
*
*      UFTEND                                     @000.000
*
*      SPACE 5
*****
*      RAMFIL - FIXED FILE AND POOL RECORD ALLOCATIONS *
*****
*
*      * START DEVICE A
*
*      * NEXT BASE ADDRESS IS 00002
*
*      *      LARGE DUPED FIXED FILE RECORDS
*      *      3828 RECORDS

```

```

*
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BASE=00002, X
      UFTI4=(62,296),UFTI5=(52,296),EQU=441,USER=(*,H,1)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,297),UFTI5=(52,297),EQU=441,USER=(*,H,2)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,298),UFTI5=(52,298),EQU=441,USER=(*,H,3)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,299),UFTI5=(52,299),EQU=441,USER=(*,H,4)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,300),UFTI5=(52,300),EQU=441,USER=(*,H,5)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,301),UFTI5=(52,301),EQU=441,USER=(*,H,6)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,302),UFTI5=(52,302),EQU=441,USER=(*,H,7)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,303),UFTI5=(52,303),EQU=441,USER=(*,H,8)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,261),UFTI5=(52,261),EQU=441,USER=(*,F,1)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,262),UFTI5=(52,262),EQU=441,USER=(*,F,2)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,263),UFTI5=(52,263),EQU=441,USER=(*,F,3)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,264),UFTI5=(52,264),EQU=441,USER=(*,F,4)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,265),UFTI5=(52,265),EQU=441,USER=(*,F,5)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,266),UFTI5=(52,266),EQU=441,USER=(*,F,6)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,267),UFTI5=(52,267),EQU=441,USER=(*,F,7)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,268),UFTI5=(52,268),EQU=441,USER=(*,F,8)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,269),UFTI5=(52,269),EQU=441,USER=(*,F,9)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,270),UFTI5=(52,270),EQU=441,USER=(*,F,10)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,271),UFTI5=(52,271),EQU=441,USER=(*,F,11)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,272),UFTI5=(52,272),EQU=441,USER=(*,F,12)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,273),UFTI5=(52,273),EQU=441,USER=(*,F,13)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,274),UFTI5=(52,274),EQU=441,USER=(*,F,14)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,275),UFTI5=(52,275),EQU=441,USER=(*,F,15)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,276),UFTI5=(52,276),EQU=441,USER=(*,F,16)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,277),UFTI5=(52,277),EQU=441,USER=(*,G,1)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,278),UFTI5=(52,278),EQU=441,USER=(*,G,2)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,279),UFTI5=(52,279),EQU=441,USER=(*,G,3)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,280),UFTI5=(52,280),EQU=441,USER=(*,G,4)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,281),UFTI5=(52,281),EQU=441,USER=(*,G,5)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,282),UFTI5=(52,282),EQU=441,USER=(*,G,6)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,283),UFTI5=(52,283),EQU=441,USER=(*,G,7)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,284),UFTI5=(52,284),EQU=441,USER=(*,G,8)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, X
      UFTI4=(62,285),UFTI5=(52,285),EQU=441,USER=(*,G,9)

```

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,286),UFTI5=(52,286),EQU=441,USER=(*,G,10)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,287),UFTI5=(52,287),EQU=441,USER=(*,G,11)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,288),UFTI5=(52,288),EQU=441,USER=(*,G,12)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,289),UFTI5=(52,289),EQU=441,USER=(*,G,13)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,290),UFTI5=(52,290),EQU=441,USER=(*,G,14)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,291),UFTI5=(52,291),EQU=441,USER=(*,G,15)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,292),UFTI5=(52,292),EQU=441,USER=(*,G,16)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0001, UFTI4=(63,351),UFTI5=(53,351),EQU=441,USER=(*,C,1)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0002, UFTI4=(63,352),UFTI5=(53,352),EQU=441,USER=(*,C,2)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0003, UFTI4=(63,353),UFTI5=(53,353),EQU=441,USER=(*,C,3)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0004, UFTI4=(63,354),UFTI5=(53,354),EQU=441,USER=(*,C,4)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0005, UFTI4=(63,355),UFTI5=(53,355),EQU=441,USER=(*,C,5)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0006, UFTI4=(63,356),UFTI5=(53,356),EQU=441,USER=(*,C,6)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0007, UFTI4=(63,357),UFTI5=(53,357),EQU=441,USER=(*,C,7)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0008, UFTI4=(63,358),UFTI5=(53,358),EQU=441,USER=(*,C,8)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0009, UFTI4=(63,359),UFTI5=(53,359),EQU=441,USER=(*,C,9)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0010, UFTI4=(63,360),UFTI5=(53,360),EQU=441,USER=(*,C,10)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0011, UFTI4=(63,361),UFTI5=(53,361),EQU=441,USER=(*,C,11)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0012, UFTI4=(63,362),UFTI5=(53,362),EQU=441,USER=(*,C,12)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0013, UFTI4=(63,363),UFTI5=(53,363),EQU=441,USER=(*,C,13)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0014, UFTI4=(63,364),UFTI5=(53,364),EQU=441,USER=(*,C,14)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0015, UFTI4=(63,365),UFTI5=(53,365),EQU=441,USER=(*,C,15)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0016, UFTI4=(63,366),UFTI5=(53,366),EQU=441,USER=(*,C,16)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0017, UFTI4=(63,367),UFTI5=(53,367),EQU=441,USER=(*,D,1)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0018, UFTI4=(63,368),UFTI5=(53,368),EQU=441,USER=(*,D,2)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0019, UFTI4=(63,369),UFTI5=(53,369),EQU=441,USER=(*,D,3)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0020, UFTI4=(63,370),UFTI5=(53,370),EQU=441,USER=(*,D,4)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0021, UFTI4=(63,371),UFTI5=(53,371),EQU=441,USER=(*,D,5)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0022, UFTI4=(63,372),UFTI5=(53,372),EQU=441,USER=(*,D,6)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0023, UFTI4=(63,373),UFTI5=(53,373),EQU=441,USER=(*,D,7)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0024, UFTI4=(63,374),UFTI5=(53,374),EQU=441,USER=(*,D,8)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0025, UFTI4=(63,375),UFTI5=(53,375),EQU=441,USER=(*,D,9)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0026, UFTI4=(63,376),UFTI5=(53,376),EQU=441,USER=(*,D,10)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0027,	X

UFTI4=(63,377),UFTI5=(53,377),EQU=441,USER=(*,D,11)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0028,	X
UFTI4=(63,378),UFTI5=(53,378),EQU=441,USER=(*,D,12)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0029,	X
UFTI4=(63,379),UFTI5=(53,379),EQU=441,USER=(*,D,13)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0030,	X
UFTI4=(63,380),UFTI5=(53,380),EQU=441,USER=(*,D,14)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0031,	X
UFTI4=(63,381),UFTI5=(53,381),EQU=441,USER=(*,D,15)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0032,	X
UFTI4=(63,382),UFTI5=(53,382),EQU=441,USER=(*,D,16)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0033,	X
UFTI4=(63,383),UFTI5=(53,383),EQU=441,USER=(*,E,1)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0034,	X
UFTI4=(63,384),UFTI5=(53,384),EQU=441,USER=(*,E,2)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0035,	X
UFTI4=(63,385),UFTI5=(53,385),EQU=441,USER=(*,E,3)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0036,	X
UFTI4=(63,386),UFTI5=(53,386),EQU=441,USER=(*,E,4)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0037,	X
UFTI4=(63,387),UFTI5=(53,387),EQU=441,USER=(*,E,5)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0038,	X
UFTI4=(63,388),UFTI5=(53,388),EQU=441,USER=(*,E,6)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0039,	X
UFTI4=(63,389),UFTI5=(53,389),EQU=441,USER=(*,E,7)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0040,	X
UFTI4=(63,390),UFTI5=(53,390),EQU=441,USER=(*,E,8)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0041,	X
UFTI4=(63,391),UFTI5=(53,391),EQU=441,USER=(*,E,9)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0042,	X
UFTI4=(63,392),UFTI5=(53,392),EQU=441,USER=(*,E,10)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0043,	X
UFTI4=(63,393),UFTI5=(53,393),EQU=441,USER=(*,E,11)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0044,	X
UFTI4=(63,394),UFTI5=(53,394),EQU=441,USER=(*,E,12)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0045,	X
UFTI4=(63,395),UFTI5=(53,395),EQU=441,USER=(*,E,13)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0046,	X
UFTI4=(63,396),UFTI5=(53,396),EQU=441,USER=(*,E,14)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0047,	X
UFTI4=(63,397),UFTI5=(53,397),EQU=441,USER=(*,E,15)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0048,	X
UFTI4=(63,398),UFTI5=(53,398),EQU=441,USER=(*,E,16)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0049,	X
UFTI4=(63,399),UFTI5=(53,399),EQU=441,USER=(*,Z,1)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0050,	X
UFTI4=(63,400),UFTI5=(53,400),EQU=441,USER=(*,Z,2)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0051,	X
UFTI4=(63,401),UFTI5=(53,401),EQU=441,USER=(*,Z,3)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0052,	X
UFTI4=(63,402),UFTI5=(53,402),EQU=441,USER=(*,Z,4)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0053,	X
UFTI4=(63,403),UFTI5=(53,403),EQU=441,USER=(*,Z,5)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0054,	X
UFTI4=(63,404),UFTI5=(53,404),EQU=441,USER=(*,Z,6)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0055,	X
UFTI4=(63,405),UFTI5=(53,405),EQU=441,USER=(*,Z,7)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0056,	X
UFTI4=(63,406),UFTI5=(53,406),EQU=441,USER=(*,Z,8)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0057,	X
UFTI4=(63,407),UFTI5=(53,407),EQU=441,USER=(*,Z,9)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0058,	X
UFTI4=(63,408),UFTI5=(53,408),EQU=441,USER=(*,Z,10)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0059,	X
UFTI4=(63,409),UFTI5=(53,409),EQU=441,USER=(*,Z,11)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0070,	X
UFTI4=(63,410),UFTI5=(53,410),EQU=441,USER=(*,Z,12)	

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0071, UFTI4=(63,411),UFTI5=(53,411),EQU=441,USER=(*,Z,13)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0072, UFTI4=(63,412),UFTI5=(53,412),EQU=441,USER=(*,Z,14)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0073, UFTI4=(63,413),UFTI5=(53,413),EQU=441,USER=(*,Z,15)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0074, UFTI4=(63,414),UFTI5=(53,414),EQU=441,USER=(*,Z,16)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0075, UFTI4=(63,415),UFTI5=(53,415),EQU=441,USER=(*,0,1)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0076, UFTI4=(63,416),UFTI5=(53,416),EQU=441,USER=(*,0,2)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0077, UFTI4=(63,417),UFTI5=(53,417),EQU=441,USER=(*,0,3)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0078, UFTI4=(63,418),UFTI5=(53,418),EQU=441,USER=(*,0,4)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0079, UFTI4=(63,419),UFTI5=(53,419),EQU=441,USER=(*,0,5)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0090, UFTI4=(63,420),UFTI5=(53,420),EQU=441,USER=(*,0,6)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0091, UFTI4=(63,421),UFTI5=(53,421),EQU=441,USER=(*,0,7)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0092, UFTI4=(63,422),UFTI5=(53,422),EQU=441,USER=(*,0,8)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0093, UFTI4=(63,423),UFTI5=(53,423),EQU=441,USER=(*,0,9)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0094, UFTI4=(63,424),UFTI5=(53,424),EQU=441,USER=(*,0,10)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0095, UFTI4=(63,425),UFTI5=(53,425),EQU=441,USER=(*,0,11)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0096, UFTI4=(63,426),UFTI5=(53,426),EQU=441,USER=(*,0,12)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0097, UFTI4=(63,427),UFTI5=(53,427),EQU=441,USER=(*,0,13)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0098, UFTI4=(63,428),UFTI5=(53,428),EQU=441,USER=(*,0,14)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0099, UFTI4=(63,429),UFTI5=(53,429),EQU=441,USER=(*,0,15)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=0100, UFTI4=(63,430),UFTI5=(53,430),EQU=441,USER=(*,0,16)	X
RAMFIL TYPE=LSA,RECNO=00020,RECID=#QA0JA,DUPE=YES,BAND=3130, UFTI4=(63,60),UFTI5=(53,60),EQU=440 @000.000	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3131, UFTI4=(63,53),UFTI5=(53,53),EQU=441,USER=(*,B,1)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3143, UFTI4=(63,54),UFTI5=(53,54),EQU=441,USER=(*,B,2)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3144, UFTI4=(63,55),UFTI5=(53,55),EQU=441,USER=(*,B,3)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3145, UFTI4=(63,56),UFTI5=(53,56),EQU=441,USER=(*,B,4)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3146, UFTI4=(63,57),UFTI5=(53,57),EQU=441,USER=(*,B,5)	X
RAMFIL TYPE=LSA,RECNO=00040,RECID=#QW0JW,DUPE=YES,BAND=3132, UFTI4=(63,58),UFTI5=(53,58),EQU=442 @000.000	X
RAMFIL TYPE=LSA,RECNO=00002,RECID=#QT0JT,DUPE=YES,BAND=3133, UFTI4=(63,59),UFTI5=(53,59),EQU=443 @000.000	X
RAMFIL TYPE=LSA,RECNO=00065,RECID=#RC8RFS,DUPE=YES,BAND=3127, UFTI4=(61,249),UFTI5=(51,249),EQU=397 @000.000	X
RAMFIL TYPE=LSA,RECNO=00584,RECID=#SONSP,DUPE=YES,BAND=3128, UFTI4=((61,250),(61,275)), UFTI5=((51,250),(51,275)),EQU=398 @000.000	X
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONCP,DUPE=YES,BAND=3100, UFTI4=((61,251),(62,21)),UFTI5=(51,251),EQU=370 @000.000	X
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONUP,DUPE=YES,BAND=3101, UFTI4=((61,252),(62,22)),UFTI5=(51,252),EQU=371 @000.000	X
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONSV,DUPE=YES,BAND=3102, UFTI4=((61,253),(62,23)),UFTI5=(51,253),EQU=372 @000.000	X



```

RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONDE,DUPE=YES,BAND=3103,      X
      UFTI4=((61,254),(62,24)),UFTI5=(51,254),EQU=373 @000.000
RAMFIL TYPE=LSA,RECNO=00147,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3172,      X
      UFTI4=(63,200),UFTI5=(53,200),EQU=441,USER=(*,B,6)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3173,      X
      UFTI4=(63,201),UFTI5=(53,201),EQU=441,USER=(*,B,7)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3174,      X
      UFTI4=(63,202),UFTI5=(53,202),EQU=441,USER=(*,B,8)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3175,      X
      UFTI4=(63,203),UFTI5=(53,203),EQU=441,USER=(*,B,9)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3176,      X
      UFTI4=(63,204),UFTI5=(53,204),EQU=441,USER=(*,B,10)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3177,      X
      UFTI4=(63,205),UFTI5=(53,205),EQU=441,USER=(*,B,11)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3178,      X
      UFTI4=(63,206),UFTI5=(53,206),EQU=441,USER=(*,B,12)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3179,      X
      UFTI4=(63,207),UFTI5=(53,207),EQU=441,USER=(*,B,13)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=3180,      X
      UFTI4=(63,208),UFTI5=(53,208),EQU=441,USER=(*,B,14)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=4036,      X
      UFTI4=(63,209),UFTI5=(53,209),EQU=441,USER=(*,B,15)
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,BAND=4037,      X
      UFTI4=(63,210),UFTI5=(53,210),EQU=441,USER=(*,B,16)
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONROLL,DUPE=YES,BAND=3108,    X
      UFTI4=((61,272),(62,26)),UFTI5=(51,272),EQU=378 @000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPM,DUPE=YES,BAND=3109,      X
      UFTI4=((61,273),(62,27)),UFTI5=(51,273),EQU=379 @000.000
RAMFIL TYPE=LSA,RECNO=00180,RECID=#GR0ZSR,DUPE=YES,BAND=3129,      X
      UFTI4=((61,274),(61,999)),UFTI5=(51,274),EQU=399,PRIOR=1
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1900,      X
      UFTI4=(61,255),UFTI5=(51,255),USER=(*,*,1) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1901,      X
      UFTI4=(61,256),UFTI5=(51,256),USER=(*,*,2) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1902,      X
      UFTI4=(61,257),UFTI5=(51,257),USER=(*,*,3) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1903,      X
      UFTI4=(61,258),UFTI5=(51,258),USER=(*,*,4) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1904,      X
      UFTI4=(61,259),UFTI5=(51,259),USER=(*,*,5) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1905,      X
      UFTI4=(61,260),UFTI5=(51,260),USER=(*,*,6) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1906,      X
      UFTI4=(61,261),UFTI5=(51,261),USER=(*,*,7) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1907,      X
      UFTI4=(61,262),UFTI5=(51,262),USER=(*,*,8) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1908,      X
      UFTI4=(61,263),UFTI5=(51,263),USER=(*,*,9) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1909,      X
      UFTI4=(61,264),UFTI5=(51,264),USER=(*,*,10) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1910,      X
      UFTI4=(61,265),UFTI5=(51,265),USER=(*,*,11) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1911,      X
      UFTI4=(61,266),UFTI5=(51,266),USER=(*,*,12) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1912,      X
      UFTI4=(61,267),UFTI5=(51,267),USER=(*,*,13) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1913,      X
      UFTI4=(61,268),UFTI5=(51,268),USER=(*,*,14) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1914,      X
      UFTI4=(61,269),UFTI5=(51,269),USER=(*,*,15) @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#ISU16,DUPE=YES,BAND=1915,      X
      UFTI4=(61,270),UFTI5=(51,270),USER=(*,*,16) @000.000

```

\*

\*\*\*\*\*

\*

\* NEXT BASE ADDRESS IS 000400

```

*
*   LARGE SHORT TERM DUPED POOL RECORDS (LST-A)
*   1 DIRECTORY   OF 660 ADDRESSES
*   5 DIRECTORIES OF 462 ADDRESSES EACH
*   2970 RECORDS TOTAL
*
  RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,BASE=00400,  X
        PSON=0,  X
        UFTI4=((00,01),(01,00),(02,00)),  X
        UFTI5=((10,00),(11,00),(12,00))
  RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,BASE=00407,  X
        PSON=462
  RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,BASE=00414,  X
        PSON=924
  RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,BASE=00506,  X
        PSON=1386
  RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,BASE=00513,  X
        PSON=1848
  RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=NO,POLID=ST,BASE=00605,  X
        PSON=2310
*
*****
*
* NEXT BASE ADDRESS IS 000700
*
*   LARGE DUPED FIXED FILE RECORDS
*   13860 RECORDS TOTAL
*
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,BASE=00700, X
        UFTI4=(61,209),UFTI5=(51,209),USER=(*,B,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,221),UFTI5=(51,221),USER=(*,C,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,222),UFTI5=(51,222),USER=(*,D,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,223),UFTI5=(51,223),USER=(*,E,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,224),UFTI5=(51,224),USER=(*,Z,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,225),UFTI5=(51,225),USER=(*,0,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,336),UFTI5=(51,336),USER=(*,F,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,337),UFTI5=(51,337),USER=(*,G,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,1000),UFTI5=(51,1000),USER=(*,H,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,1001),UFTI5=(51,1001),USER=(*,I,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,1002),UFTI5=(51,1002),USER=(*,J,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,1003),UFTI5=(51,1003),USER=(*,K,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,805),UFTI5=(51,805),USER=(*,L,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,566),UFTI5=(51,566),USER=(*,M,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,567),UFTI5=(51,567),USER=(*,N,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,568),UFTI5=(51,568),USER=(*,O,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,569),UFTI5=(51,569),USER=(*,P,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,570),UFTI5=(51,570),USER=(*,Q,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X
        UFTI4=(61,571),UFTI5=(51,571),USER=(*,R,*) 0000.000
  RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,  X

```



UFTI4=(61,572),UFTI5=(51,572),USER=(*,S,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,573),UFTI5=(51,573),USER=(*,T,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,574),UFTI5=(51,574),USER=(*,U,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,575),UFTI5=(51,575),USER=(*,V,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,576),UFTI5=(51,576),USER=(*,W,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,806),UFTI5=(51,806),USER=(*,X,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,807),UFTI5=(51,807),USER=(*,Y,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,808),UFTI5=(51,808),USER=(*,1,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,809),UFTI5=(51,809),USER=(*,2,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,810),UFTI5=(51,810),USER=(*,3,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,811),UFTI5=(51,811),USER=(*,4,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,812),UFTI5=(51,812),USER=(*,5,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00050,RECID=#DSCRU,DUPE=YES,EQU=259,	X
UFTI4=(61,813),UFTI5=(51,813),USER=(*,6,*)	@000.000
RAMFIL TYPE=LSA,RECNO=02390,RECID=SPARE,DUPE=YES	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3166,	X
UFTI4=((61,271),(62,29)),UFTI5=(51,271),EQU=377,	X
USER=(*,B,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3167,	X
UFTI4=((61,296),(62,30)),UFTI5=(51,296),EQU=377,	X
USER=(*,C,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3168,	X
UFTI4=((61,297),(62,37)),UFTI5=(51,297),EQU=377,	X
USER=(*,D,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3169,	X
UFTI4=((61,298),(62,38)),UFTI5=(51,298),EQU=377,	X
USER=(*,E,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3170,	X
UFTI4=((61,299),(62,39)),UFTI5=(51,299),EQU=377,	X
USER=(*,Z,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00325,RECID=#SONRPE,DUPE=YES,BAND=3171,	X
UFTI4=((61,300),(62,40)),UFTI5=(51,300),EQU=377,	X
USER=(*,0,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00014,RECID=#APPOP,DUPE=YES,BAND=1616,	X
UFTI4=(61,00),UFTI5=(51,00),EQU=152	
RAMFIL TYPE=LSA,RECNO=00070,RECID=#BKDRI,DUPE=YES,BAND=2920,	X
UFTI4=(61,01),UFTI5=(51,01),EQU=59	
RAMFIL TYPE=LSA,RECNO=00140,RECID=SPARE,DUPE=YES	@000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#GLOBL,DUPE=YES,EQU=39,USER=HPN2,	X
PRIOR=2	@000.000
RAMFIL TYPE=LSA,RECNO=00504,RECID=#CCEPS,DUPE=YES,BAND=1584,	X
UFTI4=(61,03),UFTI5=(51,03),EQU=99	
RAMFIL TYPE=LSA,RECNO=00010,RECID=#IDFUL,DUPE=YES,BAND=0609,USER=HPN,	X
UFTI4=(61,04),UFTI5=(51,04),EQU=154	
RAMFIL TYPE=LSA,RECNO=00010,RECID=#IDFUL,DUPE=YES,BAND=0609,USER=HPN2,X	
UFTI4=(61,162),UFTI5=(51,162),EQU=154	
RAMFIL TYPE=LSA,RECNO=00020,RECID=#IDFCL,DUPE=YES,BAND=0616,	X
UFTI4=(61,05),UFTI5=(51,05),EQU=155	
RAMFIL TYPE=LSA,RECNO=00022,RECID=#VFARD,DUPE=YES,EQU=54,PRIOR=2	
RAMFIL TYPE=LSA,RECNO=00007,RECID=SPARE,DUPE=YES	@000.000
RAMFIL TYPE=LSA,RECNO=00099,RECID=#GR3MSR,DUPE=YES,BAND=0611,	X
UFTI4=(61,07),UFTI5=(51,07),EQU=157	
RAMFIL TYPE=LSA,RECNO=00127,RECID=#SR00SR,DUPE=YES,BAND=0612,	X
UFTI4=(61,08),UFTI5=(51,08),EQU=158	
RAMFIL TYPE=LSA,RECNO=00127,RECID=#SR01SR,DUPE=YES,BAND=0613,	X
UFTI4=(61,09),UFTI5=(51,09),EQU=159	

RAMFIL TYPE=LSA,RECNO=00251,RECID=#SR0GSR,DUPE=YES,BAND=0614, UFTI4=(61,10),UFTI5=(51,10),EQU=160	X
RAMFIL TYPE=LSA,RECNO=00050,RECID=#C01RI,DUPE=YES,BAND=1648, UFTI4=(61,11),UFTI5=(51,11),EQU=115	X
RAMFIL TYPE=LSA,RECNO=00010,RECID=#GLOBL,DUPE=YES,EQU=39,USER=HPN, PRIOR=2	X 0000.000
RAMFIL TYPE=LSA,RECNO=00400,RECID=#DSCRI,DUPE=YES,BAND=1824, UFTI4=(61,13),UFTI5=(51,13),EQU=137	X
RAMFIL TYPE=LSA,RECNO=00090,RECID=#GLOBL,DUPE=YES,BAND=0608,USER=HPN, UFTI4=(61,14),UFTI5=(51,14),EQU=39	X
RAMFIL TYPE=LSA,RECNO=00090,RECID=#GLOBL,DUPE=YES,BAND=0608,USER=HPN2, UFTI4=(61,163),UFTI5=(51,163),EQU=39	X
RAMFIL TYPE=LSA,RECNO=00040,RECID=#IBMML,DUPE=YES,BAND=0306, UFTI4=(61,15),UFTI5=(51,15),EQU=24	X
RAMFIL TYPE=LSA,RECNO=00250,RECID=#STPKP,DUPE=YES,BAND=1990, UFTI4=((61,16),(62,28)),UFTI5=(51,16),EQU=91	X 0000.000
RAMFIL TYPE=LSA,RECNO=00030,RECID=#MISCL,DUPE=YES,BAND=0304, INUSE=YES, UFTI4=(61,17),UFTI5=(51,17),EQU=19	X X
RAMFIL TYPE=LSA,RECNO=00010,RECID=#DRIVER,DUPE=YES,BAND=3329, UFTI4=(61,18),UFTI5=(51,18),EQU=162	X
RAMFIL TYPE=LSA,RECNO=00407,RECID=#SSTQR,DUPE=YES,BAND=0860, UFTI4=(61,19),UFTI5=(51,19),EQU=163	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#PRORI,DUPE=YES,BAND=2936, UFTI4=(61,20),UFTI5=(51,20),EQU=82	X
RAMFIL TYPE=LSA,RECNO=00300,RECID=#PTSCK,DUPE=YES,BAND=1056, UFTI4=(61,21),UFTI5=(51,21),EQU=66	X
RAMFIL TYPE=LSA,RECNO=00072,RECID=#RCATR,DUPE=YES,BAND=1664, UFTI4=(61,22),UFTI5=(51,22),EQU=116	X
RAMFIL TYPE=LSA,RECNO=00250,RECID=#SONRI,DUPE=YES,BAND=1424, UFTI4=((61,23),(62,25)),UFTI5=(51,23),EQU=89	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#SORID,DUPE=YES,BAND=2904, UFTI4=(61,24),UFTI5=(51,24),EQU=58	X
RAMFIL TYPE=LSA,RECNO=01000,RECID=#SPARI,DUPE=YES,BAND=2144, UFTI4=((61,25),(63,00)), UFTI5=((51,25),(53,00)),EQU=144	X X
RAMFIL TYPE=LSA,RECNO=00200,RECID=#TAPID,DUPE=YES,BAND=2281, UFTI4=(61,26),UFTI5=(51,26),EQU=71	X
RAMFIL TYPE=LSA,RECNO=00007,RECID=#TIMRI,DUPE=YES,BAND=2292, UFTI4=(61,27),UFTI5=(51,27),EQU=37,PRIOR=1	X 0000.000
RAMFIL TYPE=LSA,RECNO=00100,RECID=#UATRI,DUPE=YES,BAND=0560, UFTI4=(61,28),UFTI5=(51,28),EQU=35	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#UMRRI,DUPE=YES,BAND=0848, UFTI4=(61,29),UFTI5=(51,29),EQU=53	X
RAMFIL TYPE=LSA,RECNO=00010,RECID=#VFARD,DUPE=YES,BAND=2290, UFTI4=(61,30),UFTI5=(51,30),EQU=54,PRIOR=1	X 0000.000
RAMFIL TYPE=LSA,RECNO=00064,RECID=#XSQC,DUPE=YES,BAND=2969, UFTI4=(61,31),UFTI5=(51,31),EQU=68	X
RAMFIL TYPE=LSA,RECNO=00145,RECID=#ASMSI,DUPE=YES,BAND=1872, UFTI4=(61,32),UFTI5=(51,32),EQU=138	X
RAMFIL TYPE=LSA,RECNO=00025,RECID=#TER01,DUPE=YES,BAND=1712, UFTI4=(61,33),UFTI5=(51,33),EQU=118	X
RAMFIL TYPE=LSA,RECNO=00025,RECID=#TER02,DUPE=YES,BAND=1728, UFTI4=(61,34),UFTI5=(51,34),EQU=119	X
RAMFIL TYPE=LSA,RECNO=00025,RECID=#TER03,DUPE=YES,BAND=1744, UFTI4=(61,35),UFTI5=(51,35),EQU=120	X
RAMFIL TYPE=LSA,RECNO=00025,RECID=#TER04,DUPE=YES,BAND=1760, UFTI4=(61,36),UFTI5=(51,36),EQU=121	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER05,DUPE=YES,BAND=1776, UFTI4=(61,37),UFTI5=(51,37),EQU=122	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER06,DUPE=YES,BAND=1936, UFTI4=(61,38),UFTI5=(51,38),EQU=123	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER07,DUPE=YES,BAND=1952, UFTI4=(61,39),UFTI5=(51,39),EQU=124	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER08,DUPE=YES,BAND=1968, UFTI4=(61,40),UFTI5=(51,40),EQU=125	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER09,DUPE=YES,BAND=1984,	X

```

      UFTI4=(61,41),UFTI5=(51,41),EQU=126
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER0A,DUPE=YES,BAND=2000,      X
      UFTI4=(61,42),UFTI5=(51,42),EQU=127
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER0B,DUPE=YES,BAND=2016,      X
      UFTI4=(61,43),UFTI5=(51,43),EQU=128
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER0C,DUPE=YES,BAND=2032,      X
      UFTI4=(61,44),UFTI5=(51,44),EQU=129
RAMFIL TYPE=LSA,RECNO=00005,RECID=#TER0D,DUPE=YES,BAND=2048,      X
      UFTI4=(61,45),UFTI5=(51,45),EQU=130
RAMFIL TYPE=LSA,RECNO=01236,RECID=#RCBRA,DUPE=YES,BAND=1600,      X
      UFTI4=((61,46),(63,01)),
      UFTI5=((51,46),(53,01)),EQU=113
RAMFIL TYPE=LSA,RECNO=00003,RECID=#PRORI,DUPE=YES,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0219,EQU=469, X
      UFTI4=(61,394),UFTI5=(51,394),USER=(*,B,*)
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0220,EQU=469, X
      UFTI4=(61,395),UFTI5=(51,395),USER=(*,C,*)
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0221,EQU=469, X
      UFTI4=(61,396),UFTI5=(51,396),USER=(*,D,*)
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0222,EQU=469, X
      UFTI4=(61,397),UFTI5=(51,397),USER=(*,E,*)
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0223,EQU=469, X
      UFTI4=(61,398),UFTI5=(51,398),USER=(*,Z,*)
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,BAND=0224,EQU=469, X
      UFTI4=(61,399),UFTI5=(51,399),USER=(*,0,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0225,EQU=470, X
      UFTI4=(61,400),UFTI5=(51,400),USER=(*,B,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0226,EQU=470, X
      UFTI4=(61,401),UFTI5=(51,401),USER=(*,C,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0227,EQU=470, X
      UFTI4=(61,402),UFTI5=(51,402),USER=(*,D,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0228,EQU=470, X
      UFTI4=(61,403),UFTI5=(51,403),USER=(*,E,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0229,EQU=470, X
      UFTI4=(61,404),UFTI5=(51,404),USER=(*,Z,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,BAND=0230,EQU=470, X
      UFTI4=(61,405),UFTI5=(51,405),USER=(*,0,*)
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,338),UFTI5=(51,338),USER=(*,F,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,339),UFTI5=(51,339),USER=(*,G,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,1004),UFTI5=(51,1004),USER=(*,H,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,1005),UFTI5=(51,1005),USER=(*,I,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,1006),UFTI5=(51,1006),USER=(*,J,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,1007),UFTI5=(51,1007),USER=(*,K,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,577),UFTI5=(51,577),USER=(*,L,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,578),UFTI5=(51,578),USER=(*,M,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,579),UFTI5=(51,579),USER=(*,N,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,580),UFTI5=(51,580),USER=(*,0,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,581),UFTI5=(51,581),USER=(*,P,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,582),UFTI5=(51,582),USER=(*,Q,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,583),UFTI5=(51,583),USER=(*,R,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,584),UFTI5=(51,584),USER=(*,S,*)      0000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,      X
      UFTI4=(61,585),UFTI5=(51,585),USER=(*,T,*)      0000.000

```

RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,586),UFTI5=(51,586),USER=(*,U,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,587),UFTI5=(51,587),USER=(*,V,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,588),UFTI5=(51,588),USER=(*,W,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,814),UFTI5=(51,814),USER=(*,X,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,815),UFTI5=(51,815),USER=(*,Y,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,816),UFTI5=(51,816),USER=(*,1,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,817),UFTI5=(51,817),USER=(*,2,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,818),UFTI5=(51,818),USER=(*,3,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,819),UFTI5=(51,819),USER=(*,4,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,820),UFTI5=(51,820),USER=(*,5,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00012,RECID=#RCATU,DUPE=YES,EQU=470,	X
UFTI4=(61,821),UFTI5=(51,821),USER=(*,6,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,406),UFTI5=(51,406),USER=(*,F,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,407),UFTI5=(51,407),USER=(*,G,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,1008),UFTI5=(51,1008),USER=(*,H,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,1009),UFTI5=(51,1009),USER=(*,I,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,1010),UFTI5=(51,1010),USER=(*,J,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,1011),UFTI5=(51,1011),USER=(*,K,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,589),UFTI5=(51,589),USER=(*,L,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,590),UFTI5=(51,590),USER=(*,M,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,591),UFTI5=(51,591),USER=(*,N,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,592),UFTI5=(51,592),USER=(*,O,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,593),UFTI5=(51,593),USER=(*,P,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,594),UFTI5=(51,594),USER=(*,Q,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,595),UFTI5=(51,595),USER=(*,R,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,596),UFTI5=(51,596),USER=(*,S,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,597),UFTI5=(51,597),USER=(*,T,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,598),UFTI5=(51,598),USER=(*,U,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,599),UFTI5=(51,599),USER=(*,V,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,600),UFTI5=(51,600),USER=(*,W,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,829),UFTI5=(51,829),USER=(*,X,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,822),UFTI5=(51,822),USER=(*,Y,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,823),UFTI5=(51,823),USER=(*,1,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X
UFTI4=(61,824),UFTI5=(51,824),USER=(*,2,*)	@000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,	X

```

      UFTI4=(61,825),UFTI5=(51,825),USER=(*,3,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,      X
      UFTI4=(61,826),UFTI5=(51,826),USER=(*,4,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,      X
      UFTI4=(61,827),UFTI5=(51,827),USER=(*,5,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#SATRU,DUPE=YES,EQU=469,      X
      UFTI4=(61,828),UFTI5=(51,828),USER=(*,6,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00210,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=LSA,RECNO=00192,RECID=#RC8RFS,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=LSA,RECNO=00159,RECID=SPARE,DUPE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 002100
*
*      4K DUPED FIXED FILE RECORDS
*      360 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00001,RECID=SPARE,DUPE=YES,BASE=02100      @000.000
RAMFIL TYPE=4SA,RECNO=00002,RECID=#QH0JH,DUPE=YES,BAND=4070,      X
      UFTI4=(63,350),UFTI5=(53,350),EQU=446,USER=HPN2
RAMFIL TYPE=4SA,RECNO=00004,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,      X
      UFTI4=(62,74),UFTI5=(52,74),EQU=444,USER=(*,F,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,      X
      UFTI4=(62,75),UFTI5=(52,75),EQU=444,USER=(*,G,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,      X
      UFTI4=(62,76),UFTI5=(52,76),EQU=444,USER=(*,H,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,      X
      UFTI4=(62,77),UFTI5=(52,77),EQU=444,USER=(*,I,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#HOTC,DUPE=YES,BAND=3999,      X
      UFTI4=(61,335),UFTI5=(51,335),EQU=454      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#DRIVR4K,DUPE=YES,BAND=3161,      X
      UFTI4=(61,302),UFTI5=(51,302),EQU=433      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3134,      X
      UFTI4=(63,61),UFTI5=(53,61),EQU=444,USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00002,RECID=#QG0JG,DUPE=YES,BAND=3135,      X
      UFTI4=(63,62),UFTI5=(53,62),EQU=445      @000.000
RAMFIL TYPE=4SA,RECNO=00002,RECID=#QH0JH,DUPE=YES,BAND=3136,      X
      UFTI4=(63,63),UFTI5=(53,63),EQU=446,USER=HPN      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#QZ0JZ,DUPE=YES,BAND=3137,      X
      UFTI4=(63,64),UFTI5=(53,64),EQU=447      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3138,      X
      UFTI4=(63,65),UFTI5=(53,65),EQU=444,USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3139,      X
      UFTI4=(63,66),UFTI5=(53,66),EQU=444,USER=(*,D,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3140,      X
      UFTI4=(63,67),UFTI5=(53,67),EQU=444,USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3141,      X
      UFTI4=(63,68),UFTI5=(53,68),EQU=444,USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,BAND=3142,      X
      UFTI4=(63,69),UFTI5=(53,69),EQU=444,USER=(*,0,*)
RAMFIL TYPE=4SA,RECNO=00010,RECID=#IZERO,DUPE=YES,BAND=1985,      X
      UFTI4=(61,212),UFTI5=(51,220),EQU=251      @000.000
RAMFIL TYPE=4SA,RECNO=00064,RECID=#CF2LR,DUPE=YES,BAND=1807,      X
      UFTI4=(61,213),UFTI5=(51,201),EQU=14      @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#SC1RI,DUPE=YES,BAND=2811,      X
      UFTI4=(61,193),UFTI5=(51,193),EQU=273
RAMFIL TYPE=4SA,RECNO=00030,RECID=#SC2RI,DUPE=YES,BAND=2813,      X
      UFTI4=(61,194),UFTI5=(51,194),EQU=274
RAMFIL TYPE=4SA,RECNO=00010,RECID=#CMSIT,DUPE=YES,BAND=2809,      X
      UFTI4=(61,192),UFTI5=(51,192),EQU=272
RAMFIL TYPE=4SA,RECNO=00005,RECID=#CB8HD,DUPE=YES,BAND=2161,      X
      UFTI4=(61,47),UFTI5=(51,47),EQU=164
*
*****
*

```



```

* NEXT BASE ADDRESS IS 002200
*
*      SMALL LONG TERM DUPED POOL RECORDS (SDP-A)
*      3 DIRECTORIES OF 770 ADDRESSES EACH
*      3 DIRECTORIES OF 880 ADDRESSES EACH
*      4950 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=YES,POLID=LT,BASE=02200, X
      PSON=0, X
      UFTI4=((00,02),(01,01),(02,01),(02,08)), X
      UFTI5=((10,01),(11,01),(12,01))
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=YES,POLID=LT,BASE=02207, X
      PSON=770
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=YES,POLID=LT,BASE=02300, X
      PSON=1650
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=YES,POLID=LT,BASE=02307, X
      PSON=2420
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=YES,POLID=LT,BASE=02400, X
      PSON=3300
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=YES,POLID=LT,BASE=02407, X
      PSON=4070
*
*****
*
* NEXT BASE ADDRESS IS 002500
*
*      SMALL DUPED FIXED FILE RECORDS
*      9900 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=00571,RECID=SPARE,DUPE=YES,BASE=02500 @000.000
RAMFIL TYPE=SSA,RECNO=03014,RECID=#NCBRI,DUPE=YES,PRIOR=2,EQU=150
RAMFIL TYPE=SSA,RECNO=00046,RECID=#CANRI,DUPE=YES,BAND=0112, X
      UFTI4=(61,48),UFTI5=(51,48),EQU=7
RAMFIL TYPE=SSA,RECNO=00060,RECID=#IBMMS,DUPE=YES,BAND=0658, X
      UFTI4=(61,49),UFTI5=(51,49),EQU=25
RAMFIL TYPE=SSA,RECNO=00480,RECID=#LKIBR,DUPE=YES,BAND=1488, X
      UFTI4=(61,50),UFTI5=(51,50),EQU=93
RAMFIL TYPE=SSA,RECNO=00030,RECID=#MISCS,DUPE=YES,BAND=0656, X
      INUSE=YES, X
      UFTI4=(61,51),UFTI5=(51,51),EQU=42
RAMFIL TYPE=SSA,RECNO=00010,RECID=#PTKST,DUPE=YES,BAND=1072, X
      UFTI4=(61,52),UFTI5=(51,52),EQU=67
RAMFIL TYPE=SSA,RECNO=00010,RECID=#SGFRI,DUPE=YES,BAND=2280,USER=HPN, X
      UFTI4=(61,53),UFTI5=(51,53),EQU=64
RAMFIL TYPE=SSA,RECNO=00010,RECID=#SGFRI,DUPE=YES,BAND=2280,USER=HPN2,X
      UFTI4=(61,164),UFTI5=(51,164),EQU=64
RAMFIL TYPE=SSA,RECNO=00150,RECID=#SSSRI,DUPE=YES,BAND=0080, X
      UFTI4=(61,54),UFTI5=(51,54),EQU=5
RAMFIL TYPE=SSA,RECNO=00340,RECID=#UQ1UQ,DUPE=YES,BAND=0576, X
      UFTI4=(61,55),UFTI5=(51,55),EQU=36
RAMFIL TYPE=SSA,RECNO=00031,RECID=#UT2RT,DUPE=YES,BAND=0816, X
      UFTI4=(61,56),UFTI5=(51,56),EQU=51
RAMFIL TYPE=SSA,RECNO=00002,RECID=#UV1RP,DUPE=YES,BAND=0704, X
      UFTI4=(61,57),UFTI5=(51,57),EQU=45
RAMFIL TYPE=SSA,RECNO=00012,RECID=#UX1DQ,DUPE=YES,BAND=0832, X
      UFTI4=(61,58),UFTI5=(51,58),EQU=52
RAMFIL TYPE=SSA,RECNO=00002,RECID=#XD0LS,DUPE=YES,BAND=0489, X
      UFTI4=(61,59),UFTI5=(51,59),EQU=30
RAMFIL TYPE=SSA,RECNO=01000,RECID=#XI1LD,DUPE=YES,BAND=0720, X
      UFTI4=((61,60),(63,02)), X
      UFTI5=((51,60),(53,02)),EQU=46
RAMFIL TYPE=SSA,RECNO=01000,RECID=#X01LD,DUPE=YES,BAND=0736, X
      UFTI4=((61,61),(63,03)), X
      UFTI5=((51,61),(53,03)),EQU=47
RAMFIL TYPE=SSA,RECNO=00010,RECID=#XQ1RI,DUPE=YES,BAND=0768, X
      UFTI4=(61,62),UFTI5=(51,62),EQU=48
RAMFIL TYPE=SSA,RECNO=00070,RECID=#XS0RI,DUPE=YES,BAND=0320, X

```

```

      UFTI4=(61,63),UFTI5=(51,63),EQU=20,PRIOR=1      @000.000
RAMFIL TYPE=SSA,RECNO=00011,RECID=#XS1AT,DUPE=YES,BAND=0624,      X
      UFTI4=(61,64),UFTI5=(51,64),EQU=40
RAMFIL TYPE=SSA,RECNO=00150,RECID=#XT0RI,DUPE=YES,BAND=0336,      X
      UFTI4=(61,65),UFTI5=(51,65),EQU=21
RAMFIL TYPE=SSA,RECNO=00011,RECID=#XV1XV,DUPE=YES,BAND=0784,      X
      UFTI4=(61,66),UFTI5=(51,66),EQU=49
RAMFIL TYPE=SSA,RECNO=00010,RECID=#XZ1AT,DUPE=YES,BAND=0800,      X
      UFTI4=(61,67),UFTI5=(51,67),EQU=50
RAMFIL TYPE=SSA,RECNO=00765,RECID=#QL0QL,DUPE=YES,BAND=1315,      X
      UFTI4=((61,68),(63,04)),
      UFTI5=((51,68),(53,04)),EQU=165
RAMFIL TYPE=SSA,RECNO=00078,RECID=#BTREE,DUPE=YES,BAND=2156,      X
      USER=HPN,
      UFTI4=(61,200),UFTI5=(51,200),EQU=347
RAMFIL TYPE=SSA,RECNO=00078,RECID=#BTREE,DUPE=YES,BAND=2156,      X
      USER=HPN2,
      UFTI4=(61,116),UFTI5=(51,116),EQU=347      @000.000
RAMFIL TYPE=SSA,RECNO=00070,RECID=#XS0RI,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=SSA,RECNO=01835,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00022,RECID=#BTREE,DUPE=YES,      X
      USER=HPN,PRIOR=2      @000.000
RAMFIL TYPE=SSA,RECNO=00022,RECID=#BTREE,DUPE=YES,      X
      USER=HPN2,PRIOR=2      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 003100
*
*      4K DUPED FIXED FILE RECORDS
*      1080 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RI,DUPE=YES,PRIOR=4,EQU=142,      X
      BASE=3100      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#CFREC,DUPE=YES,BAND=1617,      X
      UFTI4=(61,201),UFTI5=(51,204),EQU=277      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#DBMS4,DUPE=YES,BAND=3701,      X
      UFTI4=(61,69),UFTI5=(51,69),EQU=166      @000.000
RAMFIL TYPE=4SA,RECNO=00200,RECID=#IBMM4,DUPE=YES,BAND=3703,      @000.000X
      UFTI4=(61,70),UFTI5=(51,70),EQU=26      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#PDREC,DUPE=YES,BAND=1810,      X
      UFTI4=(61,71),UFTI5=(51,71),EQU=167
*
*****
*
* NEXT BASE ADDRESS IS 003400
*
*      LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
*      6 DIRECTORIES OF 660 ADDRESSES EACH
*      3960 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03400,      X
      PSON=0,
      UFTI4=((00,03),(01,02),(02,02),(02,09),(02,10)),      X
      UFTI5=((10,02),(11,02),(12,02))
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03410,      X
      PSON=660
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03505,      X
      PSON=1320
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03600,      X
      PSON=1980
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03610,      X
      PSON=2640
RAMFIL TYPE=LSA,RECNO=00660,RECID=POOL,DUPE=YES,POLID=LT,BASE=03705,      X
      PSON=3300
*
*****

```

```

*
* NEXT BASE ADDRESS IS 003800
*
*      SMALL SHORT TERM POOL RECORDS (SST-A)
*      3 DIRECTORIES OF 770 ADDRESSES EACH
*      3 DIRECTORIES OF 880 ADDRESSES EACH
*      4950 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,BASE=03800,   X
      PSON=0,                                                         X
      UFTI4=((00,04),(01,03),(02,03),(02,06)),                       X
      UFTI5=((10,03),(11,03),(12,03))
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=NO,POLID=ST,BASE=03807,   X
      PSON=770
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,BASE=03900,   X
      PSON=1650
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=NO,POLID=ST,BASE=03907,   X
      PSON=2420
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,BASE=04000,   X
      PSON=3300
RAMFIL TYPE=SSA,RECNO=00880,RECID=POOL,DUPE=NO,POLID=ST,BASE=04007,   X
      PSON=4070
*
*****
*
* NEXT BASE ADDRESS IS 004100
*
*      4K DUPED FIXED FILE RECORDS (VERTICALLY ALLOCATED)
*      720 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00360,RECID=#KEYPT,DUPE=YES,BAND=0777,         X
      BASE=04100,                                                     X
      UFTI4=((61,72),(62,09)),                                         X
      UFTI5=((51,72),(52,09)),EQU=60
*
*****
*
* NEXT BASE ADDRESS IS 004300
*
*      LARGE DUPED FIXED FILE RECORDS
*      1980 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=00026,RECID=#TIMRI,DUPE=YES,PRIOR=2,           X
      BASE=04300                                                       @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,          X
      UFTI4=(61,408),UFTI5=(51,408),USER=(*,F,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,          X
      UFTI4=(61,409),UFTI5=(51,409),USER=(*,G,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,          X
      UFTI4=(61,1012),UFTI5=(51,1012),USER=(*,H,*)                  @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,          X
      UFTI4=(61,1013),UFTI5=(51,1013),USER=(*,I,*)                  @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1013,EQU=333, X
      UFTI4=(61,444),UFTI5=(51,444),USER=(*,B,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1014,EQU=333, X
      UFTI4=(61,445),UFTI5=(51,445),USER=(*,C,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1015,EQU=333, X
      UFTI4=(61,446),UFTI5=(51,446),USER=(*,D,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1016,EQU=333, X
      UFTI4=(61,447),UFTI5=(51,447),USER=(*,E,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1017,EQU=333, X
      UFTI4=(61,448),UFTI5=(51,448),USER=(*,Z,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,BAND=1018,EQU=333, X
      UFTI4=(61,449),UFTI5=(51,449),USER=(*,0,*)                    @000.000
RAMFIL TYPE=LSA,RECNO=00010,RECID=#C90C9,DUPE=YES,BAND=1313,         X
      UFTI4=(61,73),UFTI5=(51,73),EQU=168
RAMFIL TYPE=LSA,RECNO=00045,RECID=#T50T5,DUPE=YES,BAND=1314,         X

```



```

      UFTI4=(61,74),UFTI5=(51,74),EQU=169
RAMFIL TYPE=LSA,RECNO=00450,RECID=#WAARI,DUPE=YES,BAND=0288,      X
      UFTI4=(61,75),UFTI5=(51,75),EQU=18
RAMFIL TYPE=LSA,RECNO=00113,RECID=#SATRI,DUPE=YES,BAND=1792,      X
      UFTI4=(61,76),UFTI5=(51,76),EQU=134
RAMFIL TYPE=LSA,RECNO=00106,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRIV,DUPE=YES,BAND=1680,      X
      UFTI4=(61,79),UFTI5=(51,79),EQU=172
RAMFIL TYPE=LSA,RECNO=00020,RECID=#QB1LD,DUPE=YES,BAND=2284,USER=HPN, X
      UFTI4=(61,80),UFTI5=(51,80),EQU=173
RAMFIL TYPE=LSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#QB1LN,DUPE=YES,BAND=2224,USER=HPN, X
      UFTI4=(61,81),UFTI5=(51,81),EQU=174
RAMFIL TYPE=LSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#RB1LD,DUPE=YES,BAND=2286,USER=HPN, X
      UFTI4=(61,82),UFTI5=(51,82),EQU=175
RAMFIL TYPE=LSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#RB1LN,DUPE=YES,BAND=2192,USER=HPN, X
      UFTI4=(61,83),UFTI5=(51,83),EQU=176
RAMFIL TYPE=LSA,RECNO=01010,RECID=SPARE,DUPE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 004600
*
* SMALL DUPED FIXED FILE RECORDS
* DEBRA / GAMMA RECORD DEFINITIONS
* 1650 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=00100,RECID=SPARE,DUPE=YES,BASE=04600 @000.000
RAMFIL TYPE=SSA,RECNO=00020,RECID=#QB1SD,DUPE=YES,BAND=2282,USER=HPN, X
      UFTI4=(61,85),UFTI5=(51,85),EQU=178
RAMFIL TYPE=SSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00020,RECID=#QB1SN,DUPE=YES,BAND=0064,USER=HPN, X
      UFTI4=(61,86),UFTI5=(51,86),EQU=179
RAMFIL TYPE=SSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00020,RECID=#RB1SD,DUPE=YES,BAND=2288,USER=HPN, X
      UFTI4=(61,87),UFTI5=(51,87),EQU=180
RAMFIL TYPE=SSA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00020,RECID=#RB1SN,DUPE=YES,BAND=1296,USER=HPN, X
      UFTI4=(61,88),UFTI5=(51,88),EQU=181
RAMFIL TYPE=SSA,RECNO=00056,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00064,RECID=#CN1ST,DUPE=YES,      X
      UFTI4=(61,132),UFTI5=(51,132),EQU=249      @000.000
RAMFIL TYPE=SSA,RECNO=00010,RECID=#RS7MS,DUPE=YES,BAND=2064,      X
      UFTI4=(61,90),UFTI5=(51,90),EQU=183
RAMFIL TYPE=SSA,RECNO=00160,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=SSA,RECNO=00010,RECID=#IDFUS,DUPE=YES,BAND=2441,USER=HPN, X
      UFTI4=(61,92),UFTI5=(51,92),EQU=185
RAMFIL TYPE=SSA,RECNO=00010,RECID=#IDFUS,DUPE=YES,BAND=2441,USER=HPN2,X
      UFTI4=(61,174),UFTI5=(51,174),EQU=185
RAMFIL TYPE=SSA,RECNO=00030,RECID=#IDFCS,DUPE=YES,BAND=0615,      X
      UFTI4=(61,93),UFTI5=(51,93),EQU=186
RAMFIL TYPE=SSA,RECNO=00002,RECID=#GR91SR,DUPE=YES,BAND=2445,      X
      UFTI4=(61,188),UFTI5=(51,188),EQU=241,      X
      USER=HPN
RAMFIL TYPE=SSA,RECNO=00002,RECID=#GR91SR,DUPE=YES,BAND=2445,      X
      UFTI4=(61,190),UFTI5=(51,190),EQU=241,      X
      USER=HPN2
RAMFIL TYPE=SSA,RECNO=00026,RECID=#GR92SR,DUPE=YES,BAND=2442,      X
      UFTI4=(61,94),UFTI5=(51,94),EQU=187,      X
      USER=HPN
RAMFIL TYPE=SSA,RECNO=00026,RECID=#GR92SR,DUPE=YES,BAND=2442,      X
      UFTI4=(61,187),UFTI5=(51,187),EQU=187,      X
      USER=HPN2
RAMFIL TYPE=SSA,RECNO=00002,RECID=#GR93SR,DUPE=YES,BAND=2446,      X
      UFTI4=(61,189),UFTI5=(51,189),EQU=242,      X

```

```

      USER=HPN
RAMFIL TYPE=SSA,RECNO=00002,RECID=#GR93SR,DUPE=YES,BAND=2446,      X
      UFTI4=(61,191),UFTI5=(51,191),EQU=242,                      X
      USER=HPN2
RAMFIL TYPE=SSA,RECNO=00026,RECID=#IR02DF,DUPE=YES,BAND=2443,      X
      UFTI4=(61,95),UFTI5=(51,95),EQU=188
RAMFIL TYPE=SSA,RECNO=00026,RECID=#IR03DF,DUPE=YES,BAND=2444,      X
      UFTI4=(61,96),UFTI5=(51,96),EQU=189
RAMFIL TYPE=SSA,RECNO=00958,RECID=SPARE,DUPE=YES
*
*****
*
* NEXT BASE ADDRESS IS 004700
*
*      4K DUPED FIXED FILE RECORDS
*      720 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00120,RECID=#TLDLR,DUPE=YES,BAND=1808,USER=HPN, X
      BASE=04700,                                                  X
      UFTI4=(61,97),UFTI5=(51,97),EQU=70
RAMFIL TYPE=4SA,RECNO=00120,RECID=#TLDLR,DUPE=YES,BAND=1808,USER=HPN2,X
      UFTI4=(61,175),UFTI5=(51,175),EQU=70
RAMFIL TYPE=4SA,RECNO=00004,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=00048,RECID=#CY2KT,DUPE=YES,BAND=4061,EQU=253, X
      UFTI4=(61,341),UFTI5=(51,341)
RAMFIL TYPE=4SA,RECNO=00048,RECID=#CY2NEW,DUPE=YES,BAND=4071,EQU=460, X
      UFTI4=(61,350),UFTI5=(51,350)
RAMFIL TYPE=4SA,RECNO=00008,RECID=#SRERKL,DUPE=YES,BAND=2820,      X
      UFTI4=(61,143),UFTI5=(51,143),EQU=191
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SREXKL,DUPE=YES,BAND=2821,      X
      UFTI4=(61,144),UFTI5=(51,144),EQU=192
RAMFIL TYPE=4SA,RECNO=00010,RECID=#PSTNEW,DUPE=YES,EQU=384,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00009,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=00017,RECID=#IPSFB,DUPE=YES,BAND=2166,      X
      UFTI4=(61,204),UFTI5=(51,205),EQU=147 @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3702,      X
      USER=(*,D,*),                                              X
      UFTI4=(61,147),UFTI5=(51,147),EQU=161 @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=2293,      X
      USER=(*,D,*),                                              X
      UFTI4=(61,148),UFTI5=(51,148),EQU=170 @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3005,      X
      USER=(*,E,*),                                              X
      UFTI4=(61,196),UFTI5=(51,196),EQU=161 @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=3010,      X
      USER=(*,E,*),                                              X
      UFTI4=(61,160),UFTI5=(51,160),EQU=170 @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3006,      X
      USER=(*,Z,*),                                              X
      UFTI4=(61,161),UFTI5=(51,161),EQU=161 @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=3011,      X
      USER=(*,Z,*),                                              X
      UFTI4=(61,184),UFTI5=(51,184),EQU=170 @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3007,      X
      USER=(*,0,*),                                              X
      UFTI4=(61,185),UFTI5=(51,185),EQU=161 @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=3012,      X
      USER=(*,0,*),                                              X
      UFTI4=(61,195),UFTI5=(51,195),EQU=170 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 004900
*
*      4K LONG TERM DUPED POOL RECORDS (4DP-A)
*      1 DIRECTORY OF 360 ADDRESSES
*      12 DIRECTORIES OF 240 ADDRESSES EACH

```

```

*      3240 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=04900, X
      PSON=0, X
      UFTI4=((00,05),(01,04),(02,04),(01,12),(01,13), X
      (02,12),(02,13)), X
      UFTI5=((10,04),(11,04),(12,04))
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=04910, X
      PSON=240
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05005, X
      PSON=480
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05100, X
      PSON=720
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05110, X
      PSON=960
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05205, X
      PSON=1200
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05300, X
      PSON=1440
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05310, X
      PSON=1680
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05405, X
      PSON=1920
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05500, X
      PSON=2160
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05510, X
      PSON=2400
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=YES,POLID=LT,BASE=05605, X
      PSON=2640
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL,DUPE=YES,POLID=LT,BASE=05700, X
      PSON=2880
*
*****
*
* NEXT BASE ADDRESS IS 005800
*
*      4K DUPED FIXED FILE RECORDS
*      DEBRA / GAMMA RECORD DEFINITIONS
*      2160 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00100,RECID=#D4KBB,DUPE=YES,BAND=4009, X
      BASE=05800, X
      UFTI4=(61,99),UFTI5=(51,99),EQU=193
RAMFIL TYPE=4SA,RECNO=00100,RECID=#D4KBC,DUPE=YES,BAND=4010, X
      UFTI4=(61,100),UFTI5=(51,100),EQU=194
RAMFIL TYPE=4SA,RECNO=00051,RECID=#MQICD,DUPE=YES,BAND=4013, X
      UFTI4=(61,101),UFTI5=(51,101),EQU=195
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2808, X
      UFTI4=(61,06),UFTI5=(51,06),EQU=156,USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00004,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,230),UFTI5=(63,230),BAND=3332, X
      USER=(*,B,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,231),UFTI5=(63,231),BAND=3333, X
      USER=(*,C,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,232),UFTI5=(63,232),BAND=3334, X
      USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,233),UFTI5=(63,233),BAND=3335, X
      USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,234),UFTI5=(63,234),BAND=3336, X
      USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#TDTDR,DUPE=YES,EQU=451, X
      UFTI4=(53,235),UFTI5=(63,235),BAND=3337, X

```

```

USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,240),UFTI5=(63,240),BAND=3338, X
USER=(*,B,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,241),UFTI5=(63,241),BAND=3339, X
USER=(*,C,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,242),UFTI5=(63,242),BAND=3340, X
USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,243),UFTI5=(63,243),BAND=3341, X
USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,244),UFTI5=(63,244),BAND=3342, X
USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=20,RECID=#IBMMP4,DUPE=YES,EQU=452, X
UFTI4=(53,245),UFTI5=(63,245),BAND=3343, X
USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#QB14D,DUPE=YES,BAND=4001,USER=HPN, X
UFTI4=(61,176),UFTI5=(51,176),EQU=198
RAMFIL TYPE=4SA,RECNO=00010,RECID=#PSTCUR,DUPE=YES,EQU=385,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRM61A,DUPE=YES,BAND=4060,EQU=252, X
UFTI4=(61,340),UFTI5=(51,340)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#QB24D,DUPE=YES,BAND=4003,USER=HPN, X
UFTI4=(61,105),UFTI5=(51,105),EQU=199
RAMFIL TYPE=4SA,RECNO=00020,RECID=#QB24D,DUPE=YES,BAND=4003,USER=HPN2,X
UFTI4=(61,177),UFTI5=(51,177),EQU=199
RAMFIL TYPE=4SA,RECNO=00020,RECID=#QB34D,DUPE=YES,BAND=4005,USER=HPN, X
UFTI4=(61,106),UFTI5=(51,106),EQU=200
RAMFIL TYPE=4SA,RECNO=00020,RECID=#QB34D,DUPE=YES,BAND=4005,USER=HPN2,X
UFTI4=(61,178),UFTI5=(51,178),EQU=200
RAMFIL TYPE=4SA,RECNO=00020,RECID=#RB14D,DUPE=YES,BAND=4002,USER=HPN, X
UFTI4=(61,107),UFTI5=(51,107),EQU=201
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRM31A8,DUPE=YES,BAND=3920, X
UFTI4=(61,457),UFTI5=(51,457),EQU=299 @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRM61A8,DUPE=YES,BAND=3921, X
UFTI4=(61,456),UFTI5=(51,456),EQU=300 @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#RB24D,DUPE=YES,BAND=4004,USER=HPN, X
UFTI4=(61,108),UFTI5=(51,108),EQU=202
RAMFIL TYPE=4SA,RECNO=00020,RECID=#RB24D,DUPE=YES,BAND=4004,USER=HPN2,X
UFTI4=(61,180),UFTI5=(51,180),EQU=202
RAMFIL TYPE=4SA,RECNO=00026,RECID=#RS4AI,DUPE=YES,BAND=4020, X
UFTI4=(61,109),UFTI5=(51,109),EQU=203
RAMFIL TYPE=4SA,RECNO=00100,RECID=#CCBRI,DUPE=YES,BAND=2810, X
UFTI4=(61,110),UFTI5=(51,110),EQU=204
RAMFIL TYPE=4SA,RECNO=00098,RECID=#NCBN4,DUPE=YES,BAND=2461, X
UFTI4=(61,197),UFTI5=(51,197),EQU=247
RAMFIL TYPE=4SA,RECNO=00144,RECID=#NCBN5,DUPE=YES,BAND=2462, X
UFTI4=(61,198),UFTI5=(51,198),EQU=248
RAMFIL TYPE=4SA,RECNO=000025,RECID=#UATRI4,DUPE=YES,BAND=4014, X
UFTI4=(61,102),UFTI5=(51,102),EQU=196 @000.000
RAMFIL TYPE=4SA,RECNO=00029,RECID=#PIDRI,DUPE=YES,BAND=4012, X
UFTI4=(61,103),UFTI5=(51,103),EQU=197 @000.000
RAMFIL TYPE=4SA,RECNO=00029,RECID=#UAPRI,DUPE=YES,BAND=0305, X
UFTI4=(61,78),UFTI5=(51,78),EQU=171 @000.000
RAMFIL TYPE=4SA,RECNO=00029,RECID=#$1PLR,DUPE=YES,BAND=0307, X
UFTI4=(61,84),UFTI5=(51,84),EQU=177 @000.000
RAMFIL TYPE=4SA,RECNO=00029,RECID=#$RGCRI,DUPE=YES,BAND=1840, X
UFTI4=(61,89),UFTI5=(51,89),EQU=182 @000.000
RAMFIL TYPE=4SA,RECNO=00200,RECID=#WGTRI4,DUPE=YES,BAND=1888, X
UFTI4=(61,02),UFTI5=(51,02),EQU=153 @000.000
RAMFIL TYPE=4SA,RECNO=00006,RECID=#PSTXNEW,DUPE=YES,EQU=278,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2921, X
UFTI4=(61,291),UFTI5=(51,291),EQU=156,USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2922, X
UFTI4=(61,292),UFTI5=(51,292),EQU=156,USER=(*,D,*)

```

```

RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2923,      X
      UFTI4=(61,293),UFTI5=(51,293),EQU=156,USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2924,      X
      UFTI4=(61,294),UFTI5=(51,294),EQU=156,USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,BAND=2925,      X
      UFTI4=(61,295),UFTI5=(51,295),EQU=156,USER=(*,0,*)
RAMFIL TYPE=4SA,RECNO=00180,RECID=#XPRG1,DUPE=YES,PRIOR=6,EQU=270
RAMFIL TYPE=4SA,RECNO=00180,RECID=#XPRG2,DUPE=YES,PRIOR=6,EQU=271
RAMFIL TYPE=4SA,RECNO=00180,RECID=#XPRG3,DUPE=YES,PRIOR=6,EQU=222
RAMFIL TYPE=4SA,RECNO=00180,RECID=#XPRG4,DUPE=YES,PRIOR=6,EQU=244
*
*****
*
* NEXT BASE ADDRESS IS 006400
*
*      4K SHORT TERM POOL RECORDS (4ST-A)
*      2 DIRECTORIES OF 600 ADDRESSES EACH
*      10 DIRECTORIES OF 240 ADDRESSES EACH
*      3600 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06400,  X
      PSON=0,      X
      UFTI4=((00,06),(01,05),(02,05),(02,11),(01,10),(01,11)),X
      UFTI5=((10,05),(11,05),(12,05))
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06410,  X
      PSON=240
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06505,  X
      PSON=480
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06600,  X
      PSON=720
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06610,  X
      PSON=960
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06705,  X
      PSON=1200
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06800,  X
      PSON=1440
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06810,  X
      PSON=1680
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=06905,  X
      PSON=1920
RAMFIL TYPE=4SA,RECNO=00240,RECID=POOL,DUPE=NO,POLID=ST,BASE=07000,  X
      PSON=2160
RAMFIL TYPE=4SA,RECNO=00600,RECID=POOL,DUPE=NO,POLID=ST,BASE=07010,  X
      PSON=2400
RAMFIL TYPE=4SA,RECNO=00600,RECID=POOL,DUPE=NO,POLID=ST,BASE=07205,  X
      PSON=3000
*
*****
*
* NEXT BASE ADDRESS IS 007400
*
*      LARGE DUPED FIXED FILE RECORDS
*      990 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=00185,RECID=#RS5DI,DUPE=YES,BAND=4028,      X
      BASE=07400,      X
      UFTI4=(61,111),UFTI5=(51,111),EQU=205
RAMFIL TYPE=LSA,RECNO=00780,RECID=#GR0ZSR,DUPE=YES,PRIOR=2      0000.000
RAMFIL TYPE=LSA,RECNO=00025,RECID=SPARE,DUPE=YES      0000.000
*
*****
*
* NEXT BASE ADDRESS IS 007500
*
*      4K DUPED FIXED FILE RECORDS
*      720 RECORDS TOTAL
*

```

```

RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3008,      X
      BASE=07500,USER=(*,B,*),      X
      UFTI4=(61,12),UFTI5=(51,12),EQU=161      @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=3013,      X
      USER=(*,B,*),      X
      UFTI4=(61,77),UFTI5=(51,77),EQU=170      @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,BAND=3009,      X
      USER=(*,C,*),      X
      UFTI4=(61,145),UFTI5=(51,145),EQU=161      @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,BAND=3014,      X
      USER=(*,C,*),      X
      UFTI4=(61,146),UFTI5=(51,146),EQU=170      @000.000
RAMFIL TYPE=4SA,RECNO=00050,RECID=#RECPD,DUPE=YES,BAND=2179,      X
      UFTI4=(61,142),UFTI5=(51,142),EQU=206
RAMFIL TYPE=4SA,RECNO=00120,RECID=#TDATR,DUPE=YES,BAND=2812,      X
      UFTI4=(61,112),UFTI5=(51,112),EQU=207
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4064,EQU=256, X
      UFTI4=(61,344),UFTI5=(51,344),USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4065,EQU=256, X
      UFTI4=(61,345),UFTI5=(51,345),USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4066,EQU=256, X
      UFTI4=(61,346),UFTI5=(51,346),USER=(*,D,*)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4067,EQU=256, X
      UFTI4=(61,347),UFTI5=(51,347),USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4068,EQU=256, X
      UFTI4=(61,348),UFTI5=(51,348),USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,BAND=4069,EQU=256, X
      UFTI4=(61,349),UFTI5=(51,349),USER=(*,0,*)
RAMFIL TYPE=4SA,RECNO=020,RECID=#STCCR,DUPE=YES,BAND=1988,      X
      UFTI4=(61,113),UFTI5=(51,113),EQU=208
RAMFIL TYPE=4SA,RECNO=150,RECID=#STPCR,DUPE=YES,BAND=1989,      X
      UFTI4=(61,114),UFTI5=(51,114),EQU=209
RAMFIL TYPE=4SA,RECNO=00100,RECID=#WGTRI,DUPE=YES,BAND=1809,      X
      UFTI4=(61,115),UFTI5=(51,115),EQU=135
*
*****
*
* NEXT BASE ADDRESS IS 007700
*
* SMALL SHORT TERM POOL RECORDS (SST-A)
* 1 DIRECTORY OF 1650 ADDRESSES
* 3 DIRECTORIES OF 3300 ADDRESSES EACH
* 11550 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=NO,POLID=ST,BASE=07700, X
      PSON=4950
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=NO,POLID=ST,BASE=07900, X
      PSON=8250
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=NO,POLID=ST,BASE=08100, X
      PSON=11550
RAMFIL TYPE=SSA,RECNO=01650,RECID=POOL,DUPE=NO,POLID=ST,BASE=08300, X
      PSON=14850
*
*****
*
* NEXT BASE ADDRESS IS 008400
*
* SMALL LONG TERM DUPED POOL RECORDS (SDP-A)
* 1 DIRECTORY OF 1650 ADDRESSES
* 3 DIRECTORIES OF 3300 ADDRESSES EACH
* 11550 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=YES,POLID=LT,BASE=08400, X
      PSON=4950
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=YES,POLID=LT,BASE=08600, X
      PSON=8250
RAMFIL TYPE=SSA,RECNO=03300,RECID=POOL,DUPE=YES,POLID=LT,BASE=08800, X

```

```

                PSON=11550
    RAMFIL TYPE=SSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,BASE=09000,  X
                PSON=14850
*
*****
*
* NEXT BASE ADDRESS IS 009100
*
*     LARGE SHORT TERM POOL RECORDS (LST-A)
*     6 DIRECTORIES OF 990 ADDRESSES EACH
*     5940 RECORDS TOTAL
*
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09100,  X
                PSON=2970
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09200,  X
                PSON=3960
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09300,  X
                PSON=4950
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09400,  X
                PSON=5940
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09500,  X
                PSON=6930
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=NO,POLID=ST,BASE=09600,  X
                PSON=7920
*
*****
*
* NEXT BASE ADDRESS IS 009700
*
*     LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
*     6 DIRECTORIES OF 990 ADDRESSES EACH
*     5940 RECORDS TOTAL
*
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=09700,  X
                PSON=3960
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=09800,  X
                PSON=4950
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=09900,  X
                PSON=5940
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=10000,  X
                PSON=6930
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=10100,  X
                PSON=7920
    RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,BASE=10200,  X
                PSON=8910
*
*****
*
* NEXT BASE ADDRESS IS 010300
*
*     LARGE DUPED FIXED FILE RECORDS
*     4950 RECORDS TOTAL
*
    RAMFIL TYPE=LSA,RECNO=01120,RECID=SPARE,DUPE=YES,BASE=10300  0000.000
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,304),UFTI5=(52,304),EQU=441,USER=(*,H,9)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,305),UFTI5=(52,305),EQU=441,USER=(*,H,10)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,306),UFTI5=(52,306),EQU=441,USER=(*,H,11)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,307),UFTI5=(52,307),EQU=441,USER=(*,H,12)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,308),UFTI5=(52,308),EQU=441,USER=(*,H,13)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X
                UFTI4=(62,309),UFTI5=(52,309),EQU=441,USER=(*,H,14)
    RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,          X

```



UFTI4=(62,310),UFTI5=(52,310),EQU=441,USER=(*,H,15)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,921),UFTI5=(52,921),EQU=441,USER=(*,H,16)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,311),UFTI5=(52,311),EQU=441,USER=(*,I,1)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,312),UFTI5=(52,312),EQU=441,USER=(*,I,2)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,313),UFTI5=(52,313),EQU=441,USER=(*,I,3)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,314),UFTI5=(52,314),EQU=441,USER=(*,I,4)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,315),UFTI5=(52,315),EQU=441,USER=(*,I,5)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,316),UFTI5=(52,316),EQU=441,USER=(*,I,6)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,317),UFTI5=(52,317),EQU=441,USER=(*,I,7)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,318),UFTI5=(52,318),EQU=441,USER=(*,I,8)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,319),UFTI5=(52,319),EQU=441,USER=(*,I,9)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,320),UFTI5=(52,320),EQU=441,USER=(*,I,10)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,321),UFTI5=(52,321),EQU=441,USER=(*,I,11)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,322),UFTI5=(52,322),EQU=441,USER=(*,I,12)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,323),UFTI5=(52,323),EQU=441,USER=(*,I,13)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,324),UFTI5=(52,324),EQU=441,USER=(*,I,14)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,325),UFTI5=(52,325),EQU=441,USER=(*,I,15)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,922),UFTI5=(52,922),EQU=441,USER=(*,I,16)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,326),UFTI5=(52,326),EQU=441,USER=(*,J,1)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,327),UFTI5=(52,327),EQU=441,USER=(*,J,2)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,328),UFTI5=(52,328),EQU=441,USER=(*,J,3)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,329),UFTI5=(52,329),EQU=441,USER=(*,J,4)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,330),UFTI5=(52,330),EQU=441,USER=(*,J,5)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,331),UFTI5=(52,331),EQU=441,USER=(*,J,6)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,332),UFTI5=(52,332),EQU=441,USER=(*,J,7)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,333),UFTI5=(52,333),EQU=441,USER=(*,J,8)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,334),UFTI5=(52,334),EQU=441,USER=(*,J,9)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,335),UFTI5=(52,335),EQU=441,USER=(*,J,10)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,336),UFTI5=(52,336),EQU=441,USER=(*,J,11)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,337),UFTI5=(52,337),EQU=441,USER=(*,J,12)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,338),UFTI5=(52,338),EQU=441,USER=(*,J,13)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,339),UFTI5=(52,339),EQU=441,USER=(*,J,14)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,340),UFTI5=(52,340),EQU=441,USER=(*,J,15)	
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,	X
UFTI4=(62,923),UFTI5=(52,923),EQU=441,USER=(*,J,16)	



RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,341),UFTI5=(52,341),EQU=441,USER=(*,K,1)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,342),UFTI5=(52,342),EQU=441,USER=(*,K,2)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,343),UFTI5=(52,343),EQU=441,USER=(*,K,3)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,344),UFTI5=(52,344),EQU=441,USER=(*,K,4)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,345),UFTI5=(52,345),EQU=441,USER=(*,K,5)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,346),UFTI5=(52,346),EQU=441,USER=(*,K,6)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,347),UFTI5=(52,347),EQU=441,USER=(*,K,7)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,348),UFTI5=(52,348),EQU=441,USER=(*,K,8)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,349),UFTI5=(52,349),EQU=441,USER=(*,K,9)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,439),UFTI5=(52,439),EQU=441,USER=(*,K,10)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,440),UFTI5=(52,440),EQU=441,USER=(*,K,11)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,441),UFTI5=(52,441),EQU=441,USER=(*,K,12)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,442),UFTI5=(52,442),EQU=441,USER=(*,K,13)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,443),UFTI5=(52,443),EQU=441,USER=(*,K,14)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,444),UFTI5=(52,444),EQU=441,USER=(*,K,15)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES, UFTI4=(62,924),UFTI5=(52,924),EQU=441,USER=(*,K,16)	X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,457),UFTI5=(52,457),USER=(*,L,1)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,458),UFTI5=(52,458),USER=(*,L,2)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,459),UFTI5=(52,459),USER=(*,L,3)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,460),UFTI5=(52,460),USER=(*,L,4)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,461),UFTI5=(52,461),USER=(*,L,5)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,462),UFTI5=(52,462),USER=(*,L,6)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,463),UFTI5=(52,463),USER=(*,L,7)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,464),UFTI5=(52,464),USER=(*,L,8)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,465),UFTI5=(52,465),USER=(*,L,9)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,466),UFTI5=(52,466),USER=(*,L,10)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,467),UFTI5=(52,467),USER=(*,L,11)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,468),UFTI5=(52,468),USER=(*,L,12)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,469),UFTI5=(52,469),USER=(*,L,13)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,470),UFTI5=(52,470),USER=(*,L,14)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,471),UFTI5=(52,471),USER=(*,L,15)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,472),UFTI5=(52,472),USER=(*,L,16)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,473),UFTI5=(52,473),USER=(*,M,1)	0000.000 X
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X

UFTI4=(62,474),UFTI5=(52,474),USER=(*,M,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,475),UFTI5=(52,475),USER=(*,M,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,476),UFTI5=(52,476),USER=(*,M,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,477),UFTI5=(52,477),USER=(*,M,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,478),UFTI5=(52,478),USER=(*,M,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,479),UFTI5=(52,479),USER=(*,M,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,480),UFTI5=(52,480),USER=(*,M,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,481),UFTI5=(52,481),USER=(*,M,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,482),UFTI5=(52,482),USER=(*,M,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,483),UFTI5=(52,483),USER=(*,M,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,484),UFTI5=(52,484),USER=(*,M,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,485),UFTI5=(52,485),USER=(*,M,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,486),UFTI5=(52,486),USER=(*,M,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,487),UFTI5=(52,487),USER=(*,M,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,488),UFTI5=(52,488),USER=(*,M,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,489),UFTI5=(52,489),USER=(*,N,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,490),UFTI5=(52,490),USER=(*,N,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,491),UFTI5=(52,491),USER=(*,N,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,492),UFTI5=(52,492),USER=(*,N,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,493),UFTI5=(52,493),USER=(*,N,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,494),UFTI5=(52,494),USER=(*,N,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,495),UFTI5=(52,495),USER=(*,N,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,496),UFTI5=(52,496),USER=(*,N,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,497),UFTI5=(52,497),USER=(*,N,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,498),UFTI5=(52,498),USER=(*,N,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,499),UFTI5=(52,499),USER=(*,N,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,500),UFTI5=(52,500),USER=(*,N,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,501),UFTI5=(52,501),USER=(*,N,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,502),UFTI5=(52,502),USER=(*,N,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,503),UFTI5=(52,503),USER=(*,N,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,504),UFTI5=(52,504),USER=(*,N,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,505),UFTI5=(52,505),USER=(*,O,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,506),UFTI5=(52,506),USER=(*,O,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,507),UFTI5=(52,507),USER=(*,O,3)	@000.000

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,508),UFTI5=(52,508),USER=(*,0,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,509),UFTI5=(52,509),USER=(*,0,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,510),UFTI5=(52,510),USER=(*,0,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,511),UFTI5=(52,511),USER=(*,0,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,512),UFTI5=(52,512),USER=(*,0,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,513),UFTI5=(52,513),USER=(*,0,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,514),UFTI5=(52,514),USER=(*,0,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,515),UFTI5=(52,515),USER=(*,0,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,516),UFTI5=(52,516),USER=(*,0,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,517),UFTI5=(52,517),USER=(*,0,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,733),UFTI5=(52,733),USER=(*,0,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,734),UFTI5=(52,734),USER=(*,0,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,735),UFTI5=(52,735),USER=(*,0,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,736),UFTI5=(52,736),USER=(*,P,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,522),UFTI5=(52,522),USER=(*,P,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,523),UFTI5=(52,523),USER=(*,P,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,524),UFTI5=(52,524),USER=(*,P,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,525),UFTI5=(52,525),USER=(*,P,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,526),UFTI5=(52,526),USER=(*,P,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,527),UFTI5=(52,527),USER=(*,P,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,528),UFTI5=(52,528),USER=(*,P,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,529),UFTI5=(52,529),USER=(*,P,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,530),UFTI5=(52,530),USER=(*,P,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,531),UFTI5=(52,531),USER=(*,P,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,532),UFTI5=(52,532),USER=(*,P,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,533),UFTI5=(52,533),USER=(*,P,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,534),UFTI5=(52,534),USER=(*,P,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,535),UFTI5=(52,535),USER=(*,P,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,536),UFTI5=(52,536),USER=(*,P,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,537),UFTI5=(52,537),USER=(*,Q,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,538),UFTI5=(52,538),USER=(*,Q,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,539),UFTI5=(52,539),USER=(*,Q,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,540),UFTI5=(52,540),USER=(*,Q,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X

UFTI4=(62,541),UFTI5=(52,541),USER=(*,Q,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,542),UFTI5=(52,542),USER=(*,Q,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,543),UFTI5=(52,543),USER=(*,Q,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,544),UFTI5=(52,544),USER=(*,Q,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,545),UFTI5=(52,545),USER=(*,Q,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,546),UFTI5=(52,546),USER=(*,Q,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,547),UFTI5=(52,547),USER=(*,Q,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,548),UFTI5=(52,548),USER=(*,Q,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,549),UFTI5=(52,549),USER=(*,Q,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,550),UFTI5=(52,550),USER=(*,Q,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,551),UFTI5=(52,551),USER=(*,Q,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,552),UFTI5=(52,552),USER=(*,Q,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,553),UFTI5=(52,553),USER=(*,R,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,554),UFTI5=(52,554),USER=(*,R,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,555),UFTI5=(52,555),USER=(*,R,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,556),UFTI5=(52,556),USER=(*,R,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,557),UFTI5=(52,557),USER=(*,R,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,558),UFTI5=(52,558),USER=(*,R,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,559),UFTI5=(52,559),USER=(*,R,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,560),UFTI5=(52,560),USER=(*,R,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,561),UFTI5=(52,561),USER=(*,R,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,562),UFTI5=(52,562),USER=(*,R,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,563),UFTI5=(52,563),USER=(*,R,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,564),UFTI5=(52,564),USER=(*,R,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,565),UFTI5=(52,565),USER=(*,R,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,566),UFTI5=(52,566),USER=(*,R,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,567),UFTI5=(52,567),USER=(*,R,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,568),UFTI5=(52,568),USER=(*,R,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,569),UFTI5=(52,569),USER=(*,S,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,570),UFTI5=(52,570),USER=(*,S,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,571),UFTI5=(52,571),USER=(*,S,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,572),UFTI5=(52,572),USER=(*,S,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,573),UFTI5=(52,573),USER=(*,S,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,574),UFTI5=(52,574),USER=(*,S,6)	@000.000

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,575),UFTI5=(52,575),USER=(*,S,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,576),UFTI5=(52,576),USER=(*,S,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,577),UFTI5=(52,577),USER=(*,S,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,578),UFTI5=(52,578),USER=(*,S,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,579),UFTI5=(52,579),USER=(*,S,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,580),UFTI5=(52,580),USER=(*,S,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,581),UFTI5=(52,581),USER=(*,S,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,582),UFTI5=(52,582),USER=(*,S,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,583),UFTI5=(52,583),USER=(*,S,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,584),UFTI5=(52,584),USER=(*,S,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,585),UFTI5=(52,585),USER=(*,T,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,586),UFTI5=(52,586),USER=(*,T,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,587),UFTI5=(52,587),USER=(*,T,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,588),UFTI5=(52,588),USER=(*,T,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,589),UFTI5=(52,589),USER=(*,T,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,590),UFTI5=(52,590),USER=(*,T,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,591),UFTI5=(52,591),USER=(*,T,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,592),UFTI5=(52,592),USER=(*,T,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,593),UFTI5=(52,593),USER=(*,T,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,594),UFTI5=(52,594),USER=(*,T,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,595),UFTI5=(52,595),USER=(*,T,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,596),UFTI5=(52,596),USER=(*,T,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,597),UFTI5=(52,597),USER=(*,T,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,598),UFTI5=(52,598),USER=(*,T,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,599),UFTI5=(52,599),USER=(*,T,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,600),UFTI5=(52,600),USER=(*,T,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,601),UFTI5=(52,601),USER=(*,U,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,602),UFTI5=(52,602),USER=(*,U,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,603),UFTI5=(52,603),USER=(*,U,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,604),UFTI5=(52,604),USER=(*,U,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,605),UFTI5=(52,605),USER=(*,U,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,606),UFTI5=(52,606),USER=(*,U,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,607),UFTI5=(52,607),USER=(*,U,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X



UFTI4=(62,608),UFTI5=(52,608),USER=(*,U,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,609),UFTI5=(52,609),USER=(*,U,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,610),UFTI5=(52,610),USER=(*,U,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,611),UFTI5=(52,611),USER=(*,U,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,612),UFTI5=(52,612),USER=(*,U,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,613),UFTI5=(52,613),USER=(*,U,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,614),UFTI5=(52,614),USER=(*,U,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,615),UFTI5=(52,615),USER=(*,U,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,616),UFTI5=(52,616),USER=(*,U,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,617),UFTI5=(52,617),USER=(*,V,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,618),UFTI5=(52,618),USER=(*,V,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,619),UFTI5=(52,619),USER=(*,V,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,620),UFTI5=(52,620),USER=(*,V,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,621),UFTI5=(52,621),USER=(*,V,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,622),UFTI5=(52,622),USER=(*,V,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,623),UFTI5=(52,623),USER=(*,V,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,624),UFTI5=(52,624),USER=(*,V,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,625),UFTI5=(52,625),USER=(*,V,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,626),UFTI5=(52,626),USER=(*,V,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,627),UFTI5=(52,627),USER=(*,V,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,628),UFTI5=(52,628),USER=(*,V,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,629),UFTI5=(52,629),USER=(*,V,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,630),UFTI5=(52,630),USER=(*,V,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,631),UFTI5=(52,631),USER=(*,V,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,632),UFTI5=(52,632),USER=(*,V,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,633),UFTI5=(52,633),USER=(*,W,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,634),UFTI5=(52,634),USER=(*,W,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,635),UFTI5=(52,635),USER=(*,W,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,636),UFTI5=(52,636),USER=(*,W,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,637),UFTI5=(52,637),USER=(*,W,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,638),UFTI5=(52,638),USER=(*,W,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,639),UFTI5=(52,639),USER=(*,W,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,640),UFTI5=(52,640),USER=(*,W,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,641),UFTI5=(52,641),USER=(*,W,9)	@000.000

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,642),UFTI5=(52,642),USER=(*,W,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,643),UFTI5=(52,643),USER=(*,W,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,644),UFTI5=(52,644),USER=(*,W,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,645),UFTI5=(52,645),USER=(*,W,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,646),UFTI5=(52,646),USER=(*,W,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,647),UFTI5=(52,647),USER=(*,W,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,648),UFTI5=(52,648),USER=(*,W,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,737),UFTI5=(52,737),USER=(*,X,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,738),UFTI5=(52,738),USER=(*,X,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,739),UFTI5=(52,739),USER=(*,X,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,740),UFTI5=(52,740),USER=(*,X,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,741),UFTI5=(52,741),USER=(*,X,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,742),UFTI5=(52,742),USER=(*,X,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,743),UFTI5=(52,743),USER=(*,X,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,744),UFTI5=(52,744),USER=(*,X,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,745),UFTI5=(52,745),USER=(*,X,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,746),UFTI5=(52,746),USER=(*,X,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,747),UFTI5=(52,747),USER=(*,X,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,748),UFTI5=(52,748),USER=(*,X,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,749),UFTI5=(52,749),USER=(*,X,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,750),UFTI5=(52,750),USER=(*,X,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,751),UFTI5=(52,751),USER=(*,X,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,752),UFTI5=(52,752),USER=(*,X,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,753),UFTI5=(52,753),USER=(*,Y,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,754),UFTI5=(52,754),USER=(*,Y,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,755),UFTI5=(52,755),USER=(*,Y,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,756),UFTI5=(52,756),USER=(*,Y,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,757),UFTI5=(52,757),USER=(*,Y,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,758),UFTI5=(52,758),USER=(*,Y,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,759),UFTI5=(52,759),USER=(*,Y,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,760),UFTI5=(52,760),USER=(*,Y,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,761),UFTI5=(52,761),USER=(*,Y,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,762),UFTI5=(52,762),USER=(*,Y,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X

UFTI4=(62,763),UFTI5=(52,763),USER=(*,Y,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,764),UFTI5=(52,764),USER=(*,Y,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,765),UFTI5=(52,765),USER=(*,Y,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,766),UFTI5=(52,766),USER=(*,Y,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,767),UFTI5=(52,767),USER=(*,Y,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,768),UFTI5=(52,768),USER=(*,Y,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,769),UFTI5=(52,769),USER=(*,1,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,770),UFTI5=(52,770),USER=(*,1,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,771),UFTI5=(52,771),USER=(*,1,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,772),UFTI5=(52,772),USER=(*,1,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,773),UFTI5=(52,773),USER=(*,1,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,774),UFTI5=(52,774),USER=(*,1,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,775),UFTI5=(52,775),USER=(*,1,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,776),UFTI5=(52,776),USER=(*,1,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,777),UFTI5=(52,777),USER=(*,1,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,778),UFTI5=(52,778),USER=(*,1,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,779),UFTI5=(52,779),USER=(*,1,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,780),UFTI5=(52,780),USER=(*,1,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,781),UFTI5=(52,781),USER=(*,1,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,782),UFTI5=(52,782),USER=(*,1,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,783),UFTI5=(52,783),USER=(*,1,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,784),UFTI5=(52,784),USER=(*,1,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,785),UFTI5=(52,785),USER=(*,2,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,786),UFTI5=(52,786),USER=(*,2,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,787),UFTI5=(52,787),USER=(*,2,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,788),UFTI5=(52,788),USER=(*,2,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,789),UFTI5=(52,789),USER=(*,2,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,790),UFTI5=(52,790),USER=(*,2,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,791),UFTI5=(52,791),USER=(*,2,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,792),UFTI5=(52,792),USER=(*,2,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,793),UFTI5=(52,793),USER=(*,2,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,794),UFTI5=(52,794),USER=(*,2,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,795),UFTI5=(52,795),USER=(*,2,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,796),UFTI5=(52,796),USER=(*,2,12)	@000.000



RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,797),UFTI5=(52,797),USER=(*,2,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,798),UFTI5=(52,798),USER=(*,2,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,799),UFTI5=(52,799),USER=(*,2,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,800),UFTI5=(52,800),USER=(*,2,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,801),UFTI5=(52,801),USER=(*,3,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,802),UFTI5=(52,802),USER=(*,3,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,803),UFTI5=(52,803),USER=(*,3,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,804),UFTI5=(52,804),USER=(*,3,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,805),UFTI5=(52,805),USER=(*,3,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,806),UFTI5=(52,806),USER=(*,3,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,807),UFTI5=(52,807),USER=(*,3,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,808),UFTI5=(52,808),USER=(*,3,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,809),UFTI5=(52,809),USER=(*,3,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,810),UFTI5=(52,810),USER=(*,3,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,811),UFTI5=(52,811),USER=(*,3,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,812),UFTI5=(52,812),USER=(*,3,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,813),UFTI5=(52,813),USER=(*,3,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,814),UFTI5=(52,814),USER=(*,3,14)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,815),UFTI5=(52,815),USER=(*,3,15)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,816),UFTI5=(52,816),USER=(*,3,16)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,817),UFTI5=(52,817),USER=(*,4,1)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,818),UFTI5=(52,818),USER=(*,4,2)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,819),UFTI5=(52,819),USER=(*,4,3)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,820),UFTI5=(52,820),USER=(*,4,4)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,821),UFTI5=(52,821),USER=(*,4,5)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,822),UFTI5=(52,822),USER=(*,4,6)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,823),UFTI5=(52,823),USER=(*,4,7)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,824),UFTI5=(52,824),USER=(*,4,8)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,825),UFTI5=(52,825),USER=(*,4,9)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,826),UFTI5=(52,826),USER=(*,4,10)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,827),UFTI5=(52,827),USER=(*,4,11)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,828),UFTI5=(52,828),USER=(*,4,12)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, UFTI4=(62,829),UFTI5=(52,829),USER=(*,4,13)	X 0000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X

UFTI4=(62,830),UFTI5=(52,830),USER=(*,4,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,831),UFTI5=(52,831),USER=(*,4,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,832),UFTI5=(52,832),USER=(*,4,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,833),UFTI5=(52,833),USER=(*,5,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,834),UFTI5=(52,834),USER=(*,5,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,835),UFTI5=(52,835),USER=(*,5,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,836),UFTI5=(52,836),USER=(*,5,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,837),UFTI5=(52,837),USER=(*,5,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,838),UFTI5=(52,838),USER=(*,5,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,839),UFTI5=(52,839),USER=(*,5,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,840),UFTI5=(52,840),USER=(*,5,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,841),UFTI5=(52,841),USER=(*,5,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,842),UFTI5=(52,842),USER=(*,5,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,843),UFTI5=(52,843),USER=(*,5,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,844),UFTI5=(52,844),USER=(*,5,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,845),UFTI5=(52,845),USER=(*,5,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,846),UFTI5=(52,846),USER=(*,5,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,847),UFTI5=(52,847),USER=(*,5,15)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,848),UFTI5=(52,848),USER=(*,5,16)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,849),UFTI5=(52,849),USER=(*,6,1)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,850),UFTI5=(52,850),USER=(*,6,2)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,851),UFTI5=(52,851),USER=(*,6,3)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,852),UFTI5=(52,852),USER=(*,6,4)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,853),UFTI5=(52,853),USER=(*,6,5)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,854),UFTI5=(52,854),USER=(*,6,6)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,855),UFTI5=(52,855),USER=(*,6,7)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,856),UFTI5=(52,856),USER=(*,6,8)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,857),UFTI5=(52,857),USER=(*,6,9)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,858),UFTI5=(52,858),USER=(*,6,10)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,859),UFTI5=(52,859),USER=(*,6,11)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,860),UFTI5=(52,860),USER=(*,6,12)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,861),UFTI5=(52,861),USER=(*,6,13)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,862),UFTI5=(52,862),USER=(*,6,14)	@000.000
RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441,	X
UFTI4=(62,863),UFTI5=(52,863),USER=(*,6,15)	@000.000

```

RAMFIL TYPE=LSA,RECNO=00005,RECID=#QS0JS,DUPE=YES,EQU=441, X
      UFTI4=(62,864),UFTI5=(52,864),USER=(*,6,16) @000.000
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRV1,DUPE=YES,BAND=2908, X
      UFTI4=(61,150),UFTI5=(51,150),EQU=334
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRV2,DUPE=YES,BAND=2909, X
      UFTI4=(61,151),UFTI5=(51,151),EQU=335
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRV3,DUPE=YES,BAND=2910, X
      UFTI4=(61,152),UFTI5=(51,152),EQU=336
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRV4,DUPE=YES,BAND=2911, X
      UFTI4=(61,153),UFTI5=(51,153),EQU=337
RAMFIL TYPE=LSA,RECNO=00100,RECID=#LDRV5,DUPE=YES,BAND=2912, X
      UFTI4=(61,154),UFTI5=(51,154),EQU=338
RAMFIL TYPE=LSA,RECNO=00010,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=LSA,RECNO=00080,RECID=#DFTST1,DUPE=YES,BAND=2460,USER=HPN,X
      UFTI4=(61,117),UFTI5=(51,117),EQU=211
RAMFIL TYPE=LSA,RECNO=00080,RECID=#DFTST1,DUPE=YES,BAND=2460, X
      USER=HPN2, X
      UFTI4=(61,182),UFTI5=(51,182),EQU=211
RAMFIL TYPE=LSA,RECNO=01240,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=LSA,RECNO=00020,RECID=#DFTST1,DUPE=YES,USER=HPN,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00020,RECID=#DFTST1,DUPE=YES,USER=HPN2,PRIOR=2
*
*****
*
* NEXT BASE ADDRESS IS 010800
*
* 4K DUPED FIXED FILE RECORDS
* 5760 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00024,RECID=SPARE,DUPE=YES,BASE=10800 @000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#XPRG1,DUPE=YES,PRIOR=7 @000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#XPRG2,DUPE=YES,PRIOR=7 @000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#XPRG3,DUPE=YES,PRIOR=7 @000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#XPRG4,DUPE=YES,PRIOR=7 @000.000
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG1,DUPE=YES,BAND=2186, X
      UFTI4=((62,10),(63,30),(63,40),(63,70),(63,71)), X
      UFTI5=((52,10),(53,30),(53,16)) @000.000
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG2,DUPE=YES,BAND=2187, X
      UFTI4=((62,13),(63,31),(63,41),(63,72),(63,73)), X
      UFTI5=((52,13),(53,31),(53,17)) @000.000
RAMFIL TYPE=4SA,RECNO=00040,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,PRIOR=2, X
      USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,PRIOR=2, X
      USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,PRIOR=2, X
      USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,PRIOR=2, X
      USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,PRIOR=2, X
      USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,PRIOR=2, X
      USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=00075,RECID=#RT1RI,DUPE=YES,PRIOR=2, X
      USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#RT2RI,DUPE=YES,PRIOR=2, X
      USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00350,RECID=#RRTRI,DUPE=YES,BAND=1920, X
      UFTI4=((61,120),(61,199)), X
      UFTI5=((51,120),(51,199)),EQU=141 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#SC1RI,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#SC2RI,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SA,RECNO=00400,RECID=#DBRRI,DUPE=YES,BAND=2240,USER=HPN, X
      UFTI4=(61,118),UFTI5=(51,118),EQU=136
RAMFIL TYPE=4SA,RECNO=00400,RECID=#DBRRI,DUPE=YES,BAND=2240,USER=HPN2,X
      UFTI4=(61,183),UFTI5=(51,183),EQU=136
RAMFIL TYPE=4SA,RECNO=00350,RECID=#RRTRI,DUPE=YES,PRIOR=2 @000.000

```

```

RAMFIL TYPE=4SA,RECNO=00096,RECID=#SRTRI,DUPE=YES,BAND=2160,      X
      UFTI4=(61,121),UFTI5=(51,121),EQU=145
RAMFIL TYPE=4SA,RECNO=00020,RECID=#HOTREC,DUPE=YES,BAND=2550,      X
      UFTI4=(61,122),UFTI5=(51,122),EQU=212
RAMFIL TYPE=4SA,RECNO=00630,RECID=#RV1RI,DUPE=YES,BAND=2112,      X
      UFTI4=((61,123),(61,149),(61,202),(61,203),(61,205)),      X
      UFTI5=((51,123),(51,149),(51,202),(51,203),(51,206))
RAMFIL TYPE=4SA,RECNO=00156,RECID=#RV2RI,DUPE=YES,BAND=2128,      X
      UFTI4=(61,124),UFTI5=(51,124),EQU=143
RAMFIL TYPE=4SA,RECNO=00080,RECID=#DFTST2,DUPE=YES,BAND=2480,      X
      USER=HPN,                                                  X
      UFTI4=(61,125),UFTI5=(51,125),EQU=213
RAMFIL TYPE=4SA,RECNO=00020,RECID=#HOTREC2,DUPE=YES,BAND=2482,      X
      UFTI4=(61,126),UFTI5=(51,126),EQU=214 @000.000
RAMFIL TYPE=4SA,RECNO=00144,RECID=#HOTREC,DUPE=YES,PRIOR=2 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 012400
*
* SMALL DUPED FIXED FILE RECORDS
* 11550 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=08036,RECID=SPARE,DUPE=YES,BASE=12400
RAMFIL TYPE=SSA,RECNO=00100,RECID=#SDRV1,DUPE=YES,BAND=2915,      X
      UFTI4=(61,155),UFTI5=(51,155),EQU=339
RAMFIL TYPE=SSA,RECNO=00100,RECID=#SDRV2,DUPE=YES,BAND=2916,      X
      UFTI4=(61,156),UFTI5=(51,156),EQU=340
RAMFIL TYPE=SSA,RECNO=00100,RECID=#SDRV3,DUPE=YES,BAND=2917,      X
      UFTI4=(61,157),UFTI5=(51,157),EQU=341
RAMFIL TYPE=SSA,RECNO=00100,RECID=#SDRV4,DUPE=YES,BAND=2918,      X
      UFTI4=(61,158),UFTI5=(51,158),EQU=342
RAMFIL TYPE=SSA,RECNO=00100,RECID=#SDRV5,DUPE=YES,BAND=2919,      X
      UFTI4=(61,159),UFTI5=(51,159),EQU=343
RAMFIL TYPE=SSA,RECNO=03014,RECID=#NCBRI,DUPE=YES,BAND=2176,      X
      UFTI4=((61,127),(63,05)),                                  X
      UFTI5=((51,127),(53,05)) @000.000
*
*****
*
* NEXT BASE ADDRESS IS 013100
*
* SMALL LONG TERM DUPED POOL RECORDS (SDP-A)
* 1 DIRECTORY OF 2500 ADDRESSES
* 10 DIRECTORIES OF 8000 ADDRESSES EACH
* 82500 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=82500,RECID=POOL,DUPE=YES,POLID=LT,BASE=13100, X
      PSON=16500
*
*****
*
* NEXT BASE ADDRESS IS 018100 (ENDS ON A PARTIAL TRACK)
*
* 4K LONG TERM DUPED POOL RECORDS (4DP-A)
* 1 DIRECTORY OF 5680 ADDRESSES
* 8 DIRECTORIES OF 8000 ADDRESSES EACH
* 69680 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=69680,RECID=POOL,DUPE=YES,POLID=LT,BASE=18100, X
      PSON=3240
*
*****
*
* NEXT BASE ADDRESS IS 037408 (BEGINS ON A PARTIAL TRACK)
*
* 4K DUPED FIXED FILE RECORDS

```

```

*      16000 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG1,DUPE=YES,PRIOR=5      @000.000
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG2,DUPE=YES,PRIOR=5      @000.000
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG3,DUPE=YES,PRIOR=5      @000.000
RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG4,DUPE=YES,PRIOR=5      @000.000
RAMFIL TYPE=4SA,RECNO=00036,RECID=#HOTREC,DUPE=YES,PRIOR=3
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4073,EQU=462, X
      UFTI4=((61,352),(61,450)),UFTI5=(51,352),USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4074,EQU=462, X
      UFTI4=((61,353),(61,451)),UFTI5=(51,353),USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4075,EQU=462, X
      UFTI4=((61,354),(61,452)),UFTI5=(51,354),USER=(*,D,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4076,EQU=462, X
      UFTI4=((61,355),(61,453)),UFTI5=(51,355),USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4077,EQU=462, X
      UFTI4=((61,356),(61,454)),UFTI5=(51,356),USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,BAND=4078,EQU=462, X
      UFTI4=((61,357),(61,455)),UFTI5=(51,357),USER=(*,0,*)
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4079,EQU=463, X
      UFTI4=(61,358),UFTI5=(51,358),USER=(*,B,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4080,EQU=463, X
      UFTI4=(61,359),UFTI5=(51,359),USER=(*,C,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4081,EQU=463, X
      UFTI4=(61,360),UFTI5=(51,360),USER=(*,D,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4082,EQU=463, X
      UFTI4=(61,361),UFTI5=(51,361),USER=(*,E,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4083,EQU=463, X
      UFTI4=(61,362),UFTI5=(51,362),USER=(*,Z,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,BAND=4084,EQU=463, X
      UFTI4=(61,363),UFTI5=(51,363),USER=(*,0,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4085,EQU=464, X
      UFTI4=(61,364),UFTI5=(51,364),USER=(*,B,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4086,EQU=464, X
      UFTI4=(61,365),UFTI5=(51,365),USER=(*,C,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4087,EQU=464, X
      UFTI4=(61,366),UFTI5=(51,366),USER=(*,D,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4088,EQU=464, X
      UFTI4=(61,367),UFTI5=(51,367),USER=(*,E,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4089,EQU=464, X
      UFTI4=(61,368),UFTI5=(51,368),USER=(*,Z,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,BAND=4090,EQU=464, X
      UFTI4=(61,369),UFTI5=(51,369),USER=(*,0,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=4091,EQU=465, X
      UFTI4=(61,370),UFTI5=(51,370),USER=(*,B,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=4092,EQU=465, X
      UFTI4=(61,371),UFTI5=(51,371),USER=(*,C,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=4093,EQU=465, X
      UFTI4=(61,372),UFTI5=(51,372),USER=(*,D,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=4094,EQU=465, X
      UFTI4=(61,373),UFTI5=(51,373),USER=(*,E,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=4095,EQU=465, X
      UFTI4=(61,374),UFTI5=(51,374),USER=(*,Z,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,BAND=0200,EQU=465, X
      UFTI4=(61,375),UFTI5=(51,375),USER=(*,0,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0201,EQU=466, X
      UFTI4=(61,376),UFTI5=(51,376),USER=(*,B,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0202,EQU=466, X
      UFTI4=(61,377),UFTI5=(51,377),USER=(*,C,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0203,EQU=466, X
      UFTI4=(61,378),UFTI5=(51,378),USER=(*,D,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0204,EQU=466, X
      UFTI4=(61,379),UFTI5=(51,379),USER=(*,E,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0205,EQU=466, X
      UFTI4=(61,380),UFTI5=(51,380),USER=(*,Z,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,BAND=0206,EQU=466, X
      UFTI4=(61,381),UFTI5=(51,381),USER=(*,0,*)      @000.000

```



```

RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0207,EQU=467, X
      UFTI4=(61,382),UFTI5=(51,382),USER=(*,B,*) @000.000
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0208,EQU=467, X
      UFTI4=(61,383),UFTI5=(51,383),USER=(*,C,*) @000.000
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0209,EQU=467, X
      UFTI4=(61,384),UFTI5=(51,384),USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0210,EQU=467, X
      UFTI4=(61,385),UFTI5=(51,385),USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0211,EQU=467, X
      UFTI4=(61,386),UFTI5=(51,386),USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,BAND=0212,EQU=467, X
      UFTI4=(61,387),UFTI5=(51,387),USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00025,RECID=#PSTXCUR,DUPE=YES, X
      UFTI4=(61,228),UFTI5=(51,228),EQU=276 @000.000
RAMFIL TYPE=4SA,RECNO=00019,RECID=#PSTXNEW,DUPE=YES, X
      UFTI4=(61,229),UFTI5=(51,229),EQU=278 @000.000
RAMFIL TYPE=4SA,RECNO=00032,RECID=#OSIT,DUPE=YES,BAND=4063,EQU=255, X
      UFTI4=(61,343),UFTI5=(51,343),PRIOR=1 @000.000
RAMFIL TYPE=4SA,RECNO=00202,RECID=#IRCMDF,DUPE=YES,BAND=4062,EQU=254, X
      UFTI4=((61,342),(63,441),(63,442),(63,446)), X
      UFTI5=((51,342),(53,441)),PRIOR=1 @000.000
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,B,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,C,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,D,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,E,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,Z,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=80,RECID=#IBMP4,DUPE=YES,USER=(*,0,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=05000,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,248),UFTI5=(63,248),BAND=4053,EQU=453, X
      USER=(*,B,*) @000.000
RAMFIL TYPE=4SA,RECNO=00400,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,249),UFTI5=(63,249),BAND=4054,EQU=453, X
      USER=(*,C,*) @000.000
RAMFIL TYPE=4SA,RECNO=00400,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,238),UFTI5=(63,238),BAND=4055,EQU=453, X
      USER=(*,D,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,239),UFTI5=(63,239),BAND=4056,EQU=453, X
      USER=(*,E,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,246),UFTI5=(63,246),BAND=4057,EQU=453, X
      USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES, X
      UFTI4=(53,247),UFTI5=(63,247),BAND=4058,EQU=453, X
      USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00450,RECID=#XPRG1,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SA,RECNO=00450,RECID=#XPRG2,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SA,RECNO=00450,RECID=#XPRG3,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SA,RECNO=00450,RECID=#XPRG4,DUPE=YES,PRIOR=3 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 041900
*
* LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
* 1 DIRECTORY OF 3560 ADDRESSES
* 5 DIRECTORIES OF 8000 ADDRESSES EACH
* 43560 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=43560,RECID=POOL,DUPE=YES,POLID=LT,BASE=41900, X
      PSON=9900
*
*****
*
* NEXT BASE ADDRESS IS 046300
*
* SMALL LONG TERM DUPED POOL RECORDS (SDP-A)

```

```

*      1 DIRECTORY   OF 6200 ADDRESSES
*      5 DIRECTORIES OF 8000 ADDRESSES EACH
*      46200 RECORDS TOTAL
*
  RAMFIL TYPE=SSA,RECNO=46200,RECID=POOL,DUPE=YES,POLID=LT,BASE=46300,  X
      PSON=99000
*
*****
*
* NEXT BASE ADDRESS IS 049100
*
*      SMALL SHORT TERM POOL RECORDS (SST-A)
*      1 DIRECTORY OF 1100 ADDRESSES
*      1 DIRECTORY OF 8000 ADDRESSES
*      9900 RECORDS TOTAL
*
  RAMFIL TYPE=SSA,RECNO=09900,RECID=POOL,DUPE=NO,POLID=ST,BASE=49100,  X
      PSON=16500
*
*****
*
* NEXT BASE ADDRESS IS 049700
*
*      LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
*      1 DIRECTORY   OF 3560 ADDRESSES
*      5 DIRECTORIES OF 8000 ADDRESSES EACH
*      43560 RECORDS TOTAL
*
  RAMFIL TYPE=LSA,RECNO=43560,RECID=POOL,DUPE=YES,POLID=LT,BASE=49700,  X
      PSON=53460
*
*****
*
* NEXT BASE ADDRESS IS 054100
*
*      4K DUPED FIXED FILE RECORDS
*      29520 RECORDS TOTAL
*
  RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG3,BASE=54100,DUPE=YES,          X
      UFTI4=(0,7),UFTI5=(0,8)                                0000.000
  RAMFIL TYPE=4SA,RECNO=01000,RECID=#XPRG4,DUPE=YES,                  X
      UFTI4=(0,9),UFTI5=(0,10)                                0000.000
  RAMFIL TYPE=4SA,RECNO=00196,RECID=#IDOTX,DUPE=YES,BAND=0859,PRIOR=1,  X
      UFTI4=((61,128),(61,998)),UFTI5=(51,128),EQU=216
  RAMFIL TYPE=4SA,RECNO=00060,RECID=#IDFU4,DUPE=YES,BAND=0850,USER=HPN, X
      UFTI4=(61,129),UFTI5=(51,129),EQU=217
  RAMFIL TYPE=4SA,RECNO=00060,RECID=#IDFU4,DUPE=YES,BAND=0850,USER=HPN2,X
      UFTI4=(61,186),UFTI5=(51,186),EQU=217
  RAMFIL TYPE=4SA,RECNO=00040,RECID=#IDFC4,DUPE=YES,BAND=0617,        X
      UFTI4=(61,130),UFTI5=(51,130),EQU=218
  RAMFIL TYPE=4SA,RECNO=00010,RECID=#MISC4,DUPE=YES,BAND=0851,        X
      UFTI4=(61,131),UFTI5=(51,131),EQU=28
  RAMFIL TYPE=4SA,RECNO=00015,RECID=#HDREC,DUPE=YES,EQU=250,          X
      UFTI4=(61,301),UFTI5=(51,301)                            0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,166),UFTI5=(51,166),USER=(*,B,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,171),UFTI5=(51,171),USER=(*,C,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,172),UFTI5=(51,172),USER=(*,D,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,173),UFTI5=(51,173),USER=(*,E,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,179),UFTI5=(51,179),USER=(*,Z,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X
      UFTI4=(61,207),UFTI5=(51,207),USER=(*,0,*)              0000.000
  RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,          X

```

```

      UFTI4=(61,420),UFTI5=(51,420),USER=(*,F,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,967),UFTI5=(51,967),USER=(*,G,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,526),UFTI5=(51,526),USER=(*,H,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,527),UFTI5=(51,527),USER=(*,I,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,528),UFTI5=(51,528),USER=(*,J,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,529),UFTI5=(51,529),USER=(*,K,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,685),UFTI5=(51,685),USER=(*,L,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,686),UFTI5=(51,686),USER=(*,M,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,687),UFTI5=(51,687),USER=(*,N,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,688),UFTI5=(51,688),USER=(*,O,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,689),UFTI5=(51,689),USER=(*,P,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,690),UFTI5=(51,690),USER=(*,Q,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,691),UFTI5=(51,691),USER=(*,R,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,692),UFTI5=(51,692),USER=(*,S,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,693),UFTI5=(51,693),USER=(*,T,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,694),UFTI5=(51,694),USER=(*,U,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,695),UFTI5=(51,695),USER=(*,V,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,696),UFTI5=(51,696),USER=(*,W,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,886),UFTI5=(51,886),USER=(*,X,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,887),UFTI5=(51,887),USER=(*,Y,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,888),UFTI5=(51,888),USER=(*,1,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,889),UFTI5=(51,889),USER=(*,2,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,890),UFTI5=(51,890),USER=(*,3,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,891),UFTI5=(51,891),USER=(*,4,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,892),UFTI5=(51,892),USER=(*,5,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00008,RECID=#PDREU,DUPE=YES,EQU=257,      X
      UFTI4=(61,893),UFTI5=(51,893),USER=(*,6,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,      X
      UFTI4=(52,254),UFTI5=(62,254),USER=(*,F,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,      X
      UFTI4=(52,255),UFTI5=(62,255),USER=(*,G,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,      X
      UFTI4=(52,445),UFTI5=(62,445),USER=(*,H,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,      X
      UFTI4=(52,446),UFTI5=(62,446),USER=(*,I,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,      X
      UFTI4=(52,447),UFTI5=(62,447),USER=(*,J,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00064,RECID=#BREATBL,DUPE=YES,PRIOR=2,EQU=387
RAMFIL TYPE=4SA,RECNO=00064,RECID=#BRL0TBL,DUPE=YES,PRIOR=2,EQU=386
RAMFIL TYPE=4SA,RECNO=00499,RECID=#IRDIDF,DUPE=YES,BAND=0853,PRIOR=1, X
      UFTI4=((61,133),(63,440)),UFTI5=((51,133),(53,440)),      X
      EQU=220      @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=SPARE,DUPE=YES      @000.000

```



RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR5,DUPE=YES,BAND=4040,	X
BASE=055006,	X
UFTI4=(62,41),UFTI5=(52,41),EQU=420	@000.000
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR6,DUPE=YES,BAND=4041,	X
BASE=055409,	X
UFTI4=(62,42),UFTI5=(52,42),EQU=421	@000.000
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR7,DUPE=YES,BAND=4042,	X
BASE=055812,	X
UFTI4=(62,43),UFTI5=(52,43),EQU=422	@000.000
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR8,DUPE=YES,BAND=4043,	X
BASE=056300,	X
UFTI4=(62,44),UFTI5=(52,44),EQU=423	@000.000
RAMFIL TYPE=4SA,RECNO=0007,RECID=SPARE,DUPE=YES	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,443),UFTI5=(51,443),USER=(*,F,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,966),UFTI5=(51,966),USER=(*,G,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,562),UFTI5=(51,562),USER=(*,H,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,563),UFTI5=(51,563),USER=(*,I,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,564),UFTI5=(51,564),USER=(*,J,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,565),UFTI5=(51,565),USER=(*,K,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,793),UFTI5=(51,793),USER=(*,L,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,794),UFTI5=(51,794),USER=(*,M,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,795),UFTI5=(51,795),USER=(*,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,796),UFTI5=(51,796),USER=(*,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,797),UFTI5=(51,797),USER=(*,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,798),UFTI5=(51,798),USER=(*,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,799),UFTI5=(51,799),USER=(*,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,800),UFTI5=(51,800),USER=(*,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,801),UFTI5=(51,801),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,802),UFTI5=(51,802),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,803),UFTI5=(51,803),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,804),UFTI5=(51,804),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,958),UFTI5=(51,958),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,959),UFTI5=(51,959),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,960),UFTI5=(51,960),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,961),UFTI5=(51,961),USER=(*,2,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,962),UFTI5=(51,962),USER=(*,3,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0213,EQU=468,	X
UFTI4=(61,388),UFTI5=(51,388),USER=(*,B,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0214,EQU=468,	X
UFTI4=(61,389),UFTI5=(51,389),USER=(*,C,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0215,EQU=468,	X
UFTI4=(61,390),UFTI5=(51,390),USER=(*,D,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0216,EQU=468,	X
UFTI4=(61,391),UFTI5=(51,391),USER=(*,E,*)	@000.000

```

RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0217,EQU=468, X
      UFTI4=(61,392),UFTI5=(51,392),USER=(*,Z,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,BAND=0218,EQU=468, X
      UFTI4=(61,393),UFTI5=(51,393),USER=(*,0,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTOT,DUPE=YES,BAND=3105, X
      UFTI4=(61,320),UFTI5=(51,320),EQU=375,USER=HPN2 @000.000
RAMFIL TYPE=4SA,RECNO=00650,RECID=#BRHIST,DUPE=YES,BAND=3118, X
      UFTI4=((61,321),(61,322)), X
      UFTI5=((51,321),(51,322)),EQU=388,USER=HPN2 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3181, X
      UFTI4=(61,324),UFTI5=(51,324),EQU=390,USER=(HPN,C,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3182, X
      UFTI4=(61,325),UFTI5=(51,325),EQU=390,USER=(HPN,D,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3183, X
      UFTI4=(61,326),UFTI5=(51,326),EQU=390,USER=(HPN,E,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3184, X
      UFTI4=(61,327),UFTI5=(51,327),EQU=390,USER=(HPN,Z,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3185, X
      UFTI4=(61,328),UFTI5=(51,328),EQU=390,USER=(HPN,0,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3186, X
      UFTI4=(61,329),UFTI5=(51,329),EQU=390,USER=(HPN2,B,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3187, X
      UFTI4=(61,330),UFTI5=(51,330),EQU=390,USER=(HPN2,C,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3188, X
      UFTI4=(61,331),UFTI5=(51,331),EQU=390,USER=(HPN2,D,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3189, X
      UFTI4=(61,332),UFTI5=(51,332),EQU=390,USER=(HPN2,E,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3190, X
      UFTI4=(61,333),UFTI5=(51,333),EQU=390,USER=(HPN2,Z,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3191, X
      UFTI4=(61,334),UFTI5=(51,334),EQU=390,USER=(HPN2,0,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3104, X
      UFTI4=(61,230),UFTI5=(51,230),EQU=374,USER=(HPN,B,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTOT,DUPE=YES,BAND=3105, X
      UFTI4=(61,231),UFTI5=(51,231),EQU=375,USER=HPN @000.000
RAMFIL TYPE=4SA,RECNO=00413,RECID=#SRM41A,DUPE=YES,BAND=3106, X
      UFTI4=(61,232),UFTI5=(51,232),EQU=376 @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRHH1P,DUPE=YES,BAND=3110, X
      UFTI4=(61,233),UFTI5=(51,233),EQU=380 @000.000
RAMFIL TYPE=4SA,RECNO=00036,RECID=#MPRECP,DUPE=YES,BAND=3111, X
      UFTI4=(61,234),UFTI5=(51,234),EQU=381 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BKDSV,DUPE=YES,BAND=3112, X
      UFTI4=(61,235),UFTI5=(51,235),EQU=382 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BKD4K,DUPE=YES,BAND=3113, X
      UFTI4=(61,236),UFTI5=(51,236),EQU=383 @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#PSTNEW,DUPE=YES,BAND=3114, X
      UFTI4=(61,237),UFTI5=(51,237),EQU=384 @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#PSTCUR,DUPE=YES,BAND=3115, X
      UFTI4=(61,238),UFTI5=(51,238),EQU=385 @000.000
RAMFIL TYPE=4SA,RECNO=00065,RECID=#BRL0TBL,DUPE=YES,BAND=3116, X
      UFTI4=(61,239),UFTI5=(51,239),EQU=386 @000.000
RAMFIL TYPE=4SA,RECNO=00065,RECID=#BREATBL,DUPE=YES,BAND=3117, X
      UFTI4=(61,240),UFTI5=(51,240),EQU=387 @000.000
RAMFIL TYPE=4SA,RECNO=00650,RECID=#BRHIST,DUPE=YES,BAND=3118, X
      UFTI4=((61,241),(61,276)), X
      UFTI5=((51,241),(51,276)),EQU=388,USER=HPN @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDCOR,DUPE=YES,BAND=3119, X
      UFTI4=(61,242),UFTI5=(51,242),EQU=389,PRIOR=1 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDSAV,DUPE=YES,BAND=3120, X
      UFTI4=(61,243),UFTI5=(51,243),EQU=390,USER=(HPN,B,*)
RAMFIL TYPE=4SA,RECNO=00413,RECID=#SRM51A,DUPE=YES,BAND=3121, X
      UFTI4=(61,244),UFTI5=(51,244),EQU=391 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDDEA,DUPE=YES,BAND=3122, X
      UFTI4=(61,245),UFTI5=(51,245),EQU=392 @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRM31A,DUPE=YES,BAND=3123, X
      UFTI4=(61,246),UFTI5=(51,246),EQU=393 @000.000
RAMFIL TYPE=4SA,RECNO=00101,RECID=#RGSTAT,DUPE=YES,BAND=3124, X

```

```

        UFTI4=(61,247),UFTI5=(51,247),EQU=394          @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SRMP1A,DUPE=YES,BAND=3125,      X
        UFTI4=(61,248),UFTI5=(51,248),EQU=395          @000.000
RAMFIL TYPE=4SA,RECNO=00501,RECID=#INODE,DUPE=YES,EQU=355,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00501,RECID=#FLOCK,DUPE=YES,EQU=357,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00234,RECID=#KSA1,DUPE=YES,BAND=2181,      X
        UFTI4=(61,136),UFTI5=(51,136),EQU=223
RAMFIL TYPE=4SA,RECNO=00006,RECID=SPARE,DUPE=YES          @000.000
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,B,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,C,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,D,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,E,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,Z,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=10,RECID=#TDTDR,DUPE=YES,USER=(*,0,*),PRIOR=2
RAMFIL TYPE=4SA,RECNO=00234,RECID=#KSA2,DUPE=YES,BAND=2182,      X
        UFTI4=(61,137),UFTI5=(51,137),EQU=224
RAMFIL TYPE=4SA,RECNO=00066,RECID=SPARE,DUPE=YES          @000.000
RAMFIL TYPE=4SA,RECNO=00234,RECID=#KSA3,DUPE=YES,BAND=2183,      X
        UFTI4=(61,138),UFTI5=(51,138),EQU=225
RAMFIL TYPE=4SA,RECNO=00066,RECID=SPARE,DUPE=YES          @000.000
RAMFIL TYPE=4SA,RECNO=00234,RECID=#KSA4,DUPE=YES,BAND=2184,      X
        UFTI4=(61,139),UFTI5=(51,139),EQU=226
RAMFIL TYPE=4SA,RECNO=00066,RECID=SPARE,DUPE=YES          @000.000
RAMFIL TYPE=4SA,RECNO=00234,RECID=#KBA,DUPE=YES,BAND=2185,      X
        UFTI4=(61,140),UFTI5=(51,140),EQU=227
RAMFIL TYPE=4SA,RECNO=00066,RECID=SPARE,DUPE=YES          @000.000
RAMFIL TYPE=4SA,RECNO=00050,RECID=#PVR1,DUPE=YES,BAND=2167,      X
        UFTI4=(61,141),UFTI5=(51,141),EQU=228
RAMFIL TYPE=4SA,RECNO=00050,RECID=#PVR2,DUPE=YES,BAND=2168,      X
        UFTI4=(63,08),UFTI5=(53,08),EQU=229
RAMFIL TYPE=4SA,RECNO=02000,RECID=#OLD1,DUPE=YES,BAND=2555,      X
        UFTI4=((61,119),(63,09)),
        UFTI5=((51,119),(53,09)),EQU=230
RAMFIL TYPE=4SA,RECNO=02000,RECID=#OLD2,DUPE=YES,BAND=2556,      X
        UFTI4=((63,10),(63,11)),
        UFTI5=((53,10),(53,11)),EQU=231
RAMFIL TYPE=4SA,RECNO=00200,RECID=#RV1RI,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SA,RECNO=00080,RECID=#DFTST2,DUPE=YES,BAND=2480,      X
        USER=HPN2,
        UFTI4=(61,91),UFTI5=(51,91),EQU=213          @000.000
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,EQU=156,      X
        UFTI4=(61,976),UFTI5=(51,976),USER=(*,F,*),
        UFTI4=(61,977),UFTI5=(51,977),USER=(*,G,*),
        UFTI4=(61,522),UFTI5=(51,522),USER=(*,H,*),
        UFTI4=(61,523),UFTI5=(51,523),USER=(*,I,*),
        UFTI4=(61,524),UFTI5=(51,524),USER=(*,J,*),
        UFTI4=(61,525),UFTI5=(51,525),USER=(*,K,*),
        UFTI4=(61,673),UFTI5=(51,673),USER=(*,L,*),
        UFTI4=(61,674),UFTI5=(51,674),USER=(*,M,*),
        UFTI4=(61,675),UFTI5=(51,675),USER=(*,N,*),
        UFTI4=(61,676),UFTI5=(51,676),USER=(*,O,*),
        UFTI4=(61,677),UFTI5=(51,677),USER=(*,P,*),
        UFTI4=(61,678),UFTI5=(51,678),USER=(*,Q,*),
        UFTI4=(61,679),UFTI5=(51,679),USER=(*,R,*),
RAMFIL TYPE=4SA,RECNO=00005,RECID=#IPRTE,DUPE=YES,EQU=156,      X

```

[illegible]

```

RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3155, X
      UFTI4=(61,285),UFTI5=(51,285),EQU=374,USER=(HPN2,B,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3156, X
      UFTI4=(61,286),UFTI5=(51,286),EQU=374,USER=(HPN2,C,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3157, X
      UFTI4=(61,287),UFTI5=(51,287),EQU=374,USER=(HPN2,D,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3158, X
      UFTI4=(61,288),UFTI5=(51,288),EQU=374,USER=(HPN2,E,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3159, X
      UFTI4=(61,289),UFTI5=(51,289),EQU=374,USER=(HPN2,Z,*)
RAMFIL TYPE=4SA,RECNO=00100,RECID=#BRIDTBL,DUPE=YES,BAND=3160, X
      UFTI4=(61,290),UFTI5=(51,290),EQU=374,USER=(HPN2,0,*)
RAMFIL TYPE=4SA,RECNO=04130,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=00050,RECID=#PVR3,DUPE=YES, X
      UFTI4=(0,13),UFTI5=(0,14),EQU=243 @000.000
RAMFIL TYPE=4SA,RECNO=00050,RECID=#PVR4,DUPE=YES, X
      UFTI4=(0,15),UFTI5=(0,16),EQU=246 @000.000
RAMFIL TYPE=4SA,RECNO=00048,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SA,RECNO=00128,RECID=#IDCF1,DUPE=YES, X
      UFTI4=(61,208),UFTI5=(51,208),EQU=258 @000.000
RAMFIL TYPE=4SA,RECNO=02000,RECID=#OLD3,DUPE=YES, X
      UFTI4=(0,17),UFTI5=(0,18),EQU=331 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 062300
*
* 4K DUPED FIXED FILE RECORDS (VERTICALLY ALLOCATED)
* 5304 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00012,RECID=#CTKX,DUPE=YES,BAND=2162, X
      BASE=062300, X
      UFTI4=(62,00),UFTI5=(52,00),EQU=232
RAMFIL TYPE=4SA,RECNO=00060,RECID=#IPL1,DUPE=YES,BAND=2390, X
      BASE=062301, X
      UFTI4=(62,01),UFTI5=(52,01),EQU=233
RAMFIL TYPE=4SA,RECNO=00060,RECID=#IPL2,DUPE=YES,BAND=2391, X
      BASE=062306, X
      UFTI4=(62,02),UFTI5=(52,02),EQU=234
RAMFIL TYPE=4SA,RECNO=00060,RECID=#IPL3,DUPE=YES,BAND=2392, X
      BASE=062311, X
      UFTI4=(62,03),UFTI5=(52,03),EQU=235
RAMFIL TYPE=4SA,RECNO=00060,RECID=#IPL4,DUPE=YES,BAND=2393, X
      BASE=062401, X
      UFTI4=(62,04),UFTI5=(52,04),EQU=236
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR1,DUPE=YES,BAND=2814, X
      BASE=062406, X
      UFTI4=(62,05),UFTI5=(52,05),EQU=237
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR2,DUPE=YES,BAND=2165, X
      BASE=062809, X
      UFTI4=(62,06),UFTI5=(52,06),EQU=238
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR3,DUPE=YES,BAND=2200, X
      BASE=063212, X
      UFTI4=(62,07),UFTI5=(52,07),EQU=239
RAMFIL TYPE=4SA,RECNO=00756,RECID=#CIMR4,DUPE=YES,BAND=2201, X
      BASE=063700, X
      UFTI4=(62,08),UFTI5=(52,08),EQU=240
*
*****
*
* NEXT BASE ADDRESS IS 064103
*
* 4K DUPED FIXED FILE RECORDS
* 9552 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=0020,RECID=SPARE,DUPE=YES,BASE=64103 @000.000
RAMFIL TYPE=4SA,RECNO=00048,RECID=#CY2CPY,DUPE=YES,BAND=4072,EQU=461, X

```



```

      UFTI4=(61,351),UFTI5=(51,351)
RAMFIL TYPE=4SA,RECNO=00200,RECID=#RV2RI,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#DBRRI,DUPE=YES,PRIOR=2,USER=HPN
RAMFIL TYPE=4SA,RECNO=00100,RECID=#DBRRI,DUPE=YES,PRIOR=2,USER=HPN2
RAMFIL TYPE=4SA,RECNO=00098,RECID=#NCBN4,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SA,RECNO=00830,RECID=#RV1RI,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SA,RECNO=00156,RECID=#RV2RI,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SA,RECNO=02000,RECID=#OLD4,DUPE=YES,                X
      UFTI4=(0,19),UFTI5=(0,20),EQU=332                        @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,978),UFTI5=(51,978),USER=(*,F,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,979),UFTI5=(51,979),USER=(*,G,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,538),UFTI5=(51,538),USER=(*,H,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,539),UFTI5=(51,539),USER=(*,I,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,540),UFTI5=(51,540),USER=(*,J,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,541),UFTI5=(51,541),USER=(*,K,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,709),UFTI5=(51,709),USER=(*,L,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,710),UFTI5=(51,710),USER=(*,M,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,711),UFTI5=(51,711),USER=(*,N,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,712),UFTI5=(51,712),USER=(*,O,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,713),UFTI5=(51,713),USER=(*,P,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,714),UFTI5=(51,714),USER=(*,Q,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,715),UFTI5=(51,715),USER=(*,R,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,716),UFTI5=(51,716),USER=(*,S,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,717),UFTI5=(51,717),USER=(*,T,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,718),UFTI5=(51,718),USER=(*,U,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,719),UFTI5=(51,719),USER=(*,V,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,720),UFTI5=(51,720),USER=(*,W,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,902),UFTI5=(51,902),USER=(*,X,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,903),UFTI5=(51,903),USER=(*,Y,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,904),UFTI5=(51,904),USER=(*,1,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,905),UFTI5=(51,905),USER=(*,2,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,906),UFTI5=(51,906),USER=(*,3,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,907),UFTI5=(51,907),USER=(*,4,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,908),UFTI5=(51,908),USER=(*,5,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#RT2RI,DUPE=YES,EQU=170,      X
      UFTI4=(61,909),UFTI5=(51,909),USER=(*,6,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467,      X
      UFTI4=(61,990),UFTI5=(51,990),USER=(*,F,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467,      X
      UFTI4=(61,991),UFTI5=(51,991),USER=(*,G,*)              @000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467,      X
      UFTI4=(61,542),UFTI5=(51,542),USER=(*,H,*)              @000.000

```

RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,543),UFTI5=(51,543),USER=(*,I,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,544),UFTI5=(51,544),USER=(*,J,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,545),UFTI5=(51,545),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,733),UFTI5=(51,733),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,734),UFTI5=(51,734),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,735),UFTI5=(51,735),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,736),UFTI5=(51,736),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,737),UFTI5=(51,737),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,738),UFTI5=(51,738),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,739),UFTI5=(51,739),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,980),UFTI5=(51,980),USER=(*,F,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,981),UFTI5=(51,981),USER=(*,G,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,546),UFTI5=(51,546),USER=(*,H,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,547),UFTI5=(51,547),USER=(*,I,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,548),UFTI5=(51,548),USER=(*,J,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,549),UFTI5=(51,549),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,745),UFTI5=(51,745),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,746),UFTI5=(51,746),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,747),UFTI5=(51,747),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,748),UFTI5=(51,748),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,749),UFTI5=(51,749),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,750),UFTI5=(51,750),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,751),UFTI5=(51,751),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,752),UFTI5=(51,752),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,753),UFTI5=(51,753),USER=(*,T,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,754),UFTI5=(51,754),USER=(*,U,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,755),UFTI5=(51,755),USER=(*,V,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,756),UFTI5=(51,756),USER=(*,W,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,926),UFTI5=(51,926),USER=(*,X,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,927),UFTI5=(51,927),USER=(*,Y,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,928),UFTI5=(51,928),USER=(*,1,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,929),UFTI5=(51,929),USER=(*,2,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464, UFTI4=(61,930),UFTI5=(51,930),USER=(*,3,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464,	X

UFTI4=(61,931),UFTI5=(51,931),USER=(*,4,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464,	X
UFTI4=(61,932),UFTI5=(51,932),USER=(*,5,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00035,RECID=#SC1RU,DUPE=YES,EQU=464,	X
UFTI4=(61,933),UFTI5=(51,933),USER=(*,6,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,992),UFTI5=(51,992),USER=(*,F,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,993),UFTI5=(51,993),USER=(*,G,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,550),UFTI5=(51,550),USER=(*,H,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,551),UFTI5=(51,551),USER=(*,I,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,552),UFTI5=(51,552),USER=(*,J,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,553),UFTI5=(51,553),USER=(*,K,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,757),UFTI5=(51,757),USER=(*,L,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,758),UFTI5=(51,758),USER=(*,M,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,759),UFTI5=(51,759),USER=(*,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,760),UFTI5=(51,760),USER=(*,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,761),UFTI5=(51,761),USER=(*,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,762),UFTI5=(51,762),USER=(*,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,763),UFTI5=(51,763),USER=(*,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,764),UFTI5=(51,764),USER=(*,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,765),UFTI5=(51,765),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,766),UFTI5=(51,766),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,767),UFTI5=(51,767),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,768),UFTI5=(51,768),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,934),UFTI5=(51,934),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,935),UFTI5=(51,935),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,936),UFTI5=(51,936),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,937),UFTI5=(51,937),USER=(*,2,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,938),UFTI5=(51,938),USER=(*,3,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,939),UFTI5=(51,939),USER=(*,4,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,940),UFTI5=(51,940),USER=(*,5,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#SC2RU,DUPE=YES,EQU=465,	X
UFTI4=(61,941),UFTI5=(51,941),USER=(*,6,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463,	X
UFTI4=(61,994),UFTI5=(51,994),USER=(*,F,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463,	X
UFTI4=(61,995),UFTI5=(51,995),USER=(*,G,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463,	X
UFTI4=(61,554),UFTI5=(51,554),USER=(*,H,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463,	X
UFTI4=(61,555),UFTI5=(51,555),USER=(*,I,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463,	X
UFTI4=(61,556),UFTI5=(51,556),USER=(*,J,*)	@000.000



RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,557),UFTI5=(51,557),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,769),UFTI5=(51,769),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,770),UFTI5=(51,770),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,771),UFTI5=(51,771),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,772),UFTI5=(51,772),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,773),UFTI5=(51,773),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,774),UFTI5=(51,774),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,775),UFTI5=(51,775),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,776),UFTI5=(51,776),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,777),UFTI5=(51,777),USER=(*,T,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,778),UFTI5=(51,778),USER=(*,U,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,779),UFTI5=(51,779),USER=(*,V,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,780),UFTI5=(51,780),USER=(*,W,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,942),UFTI5=(51,942),USER=(*,X,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,943),UFTI5=(51,943),USER=(*,Y,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,944),UFTI5=(51,944),USER=(*,1,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,945),UFTI5=(51,945),USER=(*,2,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,946),UFTI5=(51,946),USER=(*,3,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,947),UFTI5=(51,947),USER=(*,4,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,948),UFTI5=(51,948),USER=(*,5,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00016,RECID=#SRTRU,DUPE=YES,EQU=463, UFTI4=(61,949),UFTI5=(51,949),USER=(*,6,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,972),UFTI5=(51,972),USER=(*,F,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,973),UFTI5=(51,973),USER=(*,G,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,514),UFTI5=(51,514),USER=(*,H,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,515),UFTI5=(51,515),USER=(*,I,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,516),UFTI5=(51,516),USER=(*,J,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,517),UFTI5=(51,517),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,661),UFTI5=(51,661),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,662),UFTI5=(51,662),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,663),UFTI5=(51,663),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,664),UFTI5=(51,664),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,665),UFTI5=(51,665),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466, UFTI4=(61,666),UFTI5=(51,666),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X

UFTI4=(61,667),UFTI5=(51,667),USER=(*,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,668),UFTI5=(51,668),USER=(*,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,669),UFTI5=(51,669),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,670),UFTI5=(51,670),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,671),UFTI5=(51,671),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,672),UFTI5=(51,672),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,670),UFTI5=(51,670),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,671),UFTI5=(51,671),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,672),UFTI5=(51,672),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,673),UFTI5=(51,673),USER=(*,2,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,674),UFTI5=(51,674),USER=(*,3,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,675),UFTI5=(51,675),USER=(*,4,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,676),UFTI5=(51,676),USER=(*,5,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#CCBRU,DUPE=YES,EQU=466,	X
UFTI4=(61,677),UFTI5=(51,677),USER=(*,6,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,963),UFTI5=(51,963),USER=(*,4,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,964),UFTI5=(51,964),USER=(*,5,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#WGTRU,DUPE=YES,EQU=468,	X
UFTI4=(61,965),UFTI5=(51,965),USER=(*,6,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00572,RECID=#IDOTX,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,653),UFTI5=(62,653),USER=(HPN,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,654),UFTI5=(62,654),USER=(HPN2,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,655),UFTI5=(62,655),USER=(HPN,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,656),UFTI5=(62,656),USER=(HPN2,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,657),UFTI5=(62,657),USER=(HPN,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,658),UFTI5=(62,658),USER=(HPN2,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,659),UFTI5=(62,659),USER=(HPN,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,660),UFTI5=(62,660),USER=(HPN2,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,661),UFTI5=(62,661),USER=(HPN,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,662),UFTI5=(62,662),USER=(HPN2,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,663),UFTI5=(62,663),USER=(HPN,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,664),UFTI5=(62,664),USER=(HPN2,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,665),UFTI5=(62,665),USER=(HPN,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,666),UFTI5=(62,666),USER=(HPN2,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,667),UFTI5=(62,667),USER=(HPN,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(52,668),UFTI5=(62,668),USER=(HPN2,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450,	X

```

      UFTI4=(52,669),UFTI5=(62,669),USER=(HPN,V,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,670),UFTI5=(62,670),USER=(HPN2,V,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,671),UFTI5=(62,671),USER=(HPN,W,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,672),UFTI5=(62,672),USER=(HPN2,W,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,865),UFTI5=(62,865),USER=(HPN,X,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,866),UFTI5=(62,866),USER=(HPN2,X,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,867),UFTI5=(62,867),USER=(HPN,Y,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,868),UFTI5=(62,868),USER=(HPN2,Y,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,869),UFTI5=(62,869),USER=(HPN,1,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,870),UFTI5=(62,870),USER=(HPN2,1,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,871),UFTI5=(62,871),USER=(HPN,2,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,872),UFTI5=(62,872),USER=(HPN2,2,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,873),UFTI5=(62,873),USER=(HPN,3,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,874),UFTI5=(62,874),USER=(HPN2,3,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,875),UFTI5=(62,875),USER=(HPN,4,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,876),UFTI5=(62,876),USER=(HPN2,4,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,877),UFTI5=(62,877),USER=(HPN,5,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,878),UFTI5=(62,878),USER=(HPN2,5,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,879),UFTI5=(62,879),USER=(HPN,6,*) 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, X
      UFTI4=(52,880),UFTI5=(62,880),USER=(HPN2,6,*) 0000.000
RAMFIL TYPE=4SA,RECNO=0014,RECID=SPARE,DUPE=YES 0000.000
*
*****
*
* NEXT BASE ADDRESS IS 066711
*
* LARGE DUPED FIXED FILE RECORDS
* 7392 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=07392,RECID=SPARE,DUPE=YES,BASE=66711 0000.000
*
*****
*
* NEXT BASE ADDRESS IS 067503
*
* 4K DUPED FIXED FILE RECORDS
* 4968 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=00013,RECID=SPARE,DUPE=YES,BASE=67503 0000.000
RAMFIL TYPE=4SA,RECNO=0200,RECID=#IRCBDF,DUPE=YES,BAND=3611, X
      UFTI4=(62,221),UFTI5=(52,220),EQU=411 0000.000
RAMFIL TYPE=4SA,RECNO=1497,RECID=#IRCFDF,DUPE=YES,BAND=3612,PRIOR=1, X
      UFTI4=((62,222),(63,443)), X
      UFTI5=((52,221),(52,227)),EQU=412 0000.000
RAMFIL TYPE=4SA,RECNO=1497,RECID=#IRCGDF,DUPE=YES,BAND=3613,PRIOR=1, X
      UFTI4=((62,223),(63,444)), X
      UFTI5=((52,222),(52,228)),EQU=413 0000.000
RAMFIL TYPE=4SA,RECNO=0032,RECID=#IRCHDF,DUPE=YES,BAND=3614, X

```

```

      UFTI4=(62,224),UFTI5=(52,223),EQU=414          @000.000
RAMFIL TYPE=4SA,RECNO=1497,RECID=#IRCIDF,DUPE=YES,BAND=3615,PRIOR=1,  X
      UFTI4=((62,225),(63,445)),                      X
      UFTI5=((52,224),(52,229)),EQU=415              @000.000
RAMFIL TYPE=4SA,RECNO=0032,RECID=#IRCJDF,DUPE=YES,BAND=3616,        X
      UFTI4=(62,226),UFTI5=(52,225),EQU=416          @000.000
RAMFIL TYPE=4SA,RECNO=0200,RECID=#IRCKDF,DUPE=YES,BAND=3617,        X
      UFTI4=(62,227),UFTI5=(52,226),EQU=417          @000.000
*
*****
*
* NEXT BASE ADDRESS IS 068900
*
*      4K LONG TERM DUPED POOL RECORDS (4DP-A)
*      1 DIRECTORY OF 4000 ADDRESSES
*      4 DIRECTORIES OF 8000 ADDRESSES EACH
*      36000 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=36000,RECID=POOL,DUPE=YES,POLID=LT,BASE=68900,  X
      PSON=88920
*
*****
*
* NEXT BASE ADDRESS IS 078900
*
*      LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
*      1 DIRECTORY OF 7600 ADDRESSES
*      4 DIRECTORIES OF 8000 ADDRESSES EACH
*      39600 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=39600,RECID=POOL,DUPE=YES,POLID=LT,BASE=78900,  X
      PSON=97020
*
*****
*
* NEXT BASE ADDRESS IS 082900
*
*      LARGE SHORT TERM POOL RECORDS (LST-A)
*      1 DIRECTORY OF 2890 ADDRESSES
*      1 DIRECTORY OF 8000 ADDRESSES
*      10890 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=10890,RECID=POOL,DUPE=NO,POLID=ST,BASE=82900,   X
      PSON=8910
*
*****
*
* NEXT BASE ADDRESS IS 084000
*
*      4K SHORT TERM POOL RECORDS (4ST-A)
*      1 DIRECTORY OF 6400 ADDRESSES
*      1 DIRECTORY OF 8000 ADDRESSES
*      14400 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=14400,RECID=POOL,DUPE=NO,POLID=ST,BASE=84000,   X
      PSON=3600
*
*****
*
* NEXT BASE ADDRESS IS 088000
*
*      SMALL SHORT TERM POOL RECORDS (SST-A)
*      1 DIRECTORY OF 5000 ADDRESSES
*      20 DIRECTORIES OF 8000 ADDRESSES
*      165000 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=165000,RECID=POOL,DUPE=NO,POLID=ST,BASE=88000,  X

```

```

                PSON=26400
*
*****
*
* NEXT BASE ADDRESS IS 098000
*
*       LARGE SHORT TERM POOL RECORDS (LST-A)
*       1 DIRECTORY OF 1500 ADDRESSES
*       6 DIRECTORIES OF 8000 ADDRESSES
*       49500 RECORDS TOTAL
*
*       RAMFIL TYPE=LSA,RECNO=49500,RECID=POOL,DUPE=NO,POLID=ST,BASE=98000,    X
                PSON=19800
*
*****
*
* NEXT BASE ADDRESS IS 103000
*
*       4K SHORT TERM POOL RECORDS (4ST-A)
*       1 DIRECTORY OF 7848 ADDRESSES
*       7848 RECORDS TOTAL
*
*       RAMFIL TYPE=4SA,RECNO=07848,RECID=POOL,DUPE=NO,POLID=ST,BASE=103000,  X
                PSON=18000
*
*****
*
* NEXT BASE ADDRESS IS 105112
*
*       4K DUPED FIXED FILE RECORDS
*       20952 RECORDS TOTAL
*
*       RAMFIL TYPE=4SA,RECNO=01500,RECID=#INODE,DUPE=YES,BAND=3705,          X
                UFTI4=(63,20),UFTI5=(51,210),                                X
                BASE=105112                                                  @000.000
*       RAMFIL TYPE=4SA,RECNO=00100,RECID=#TPFCWD,DUPE=YES,BAND=3706,          X
                UFTI4=(63,21),UFTI5=(51,211),EQU=356                      @000.000
*       RAMFIL TYPE=4SA,RECNO=01500,RECID=#FLOCK,DUPE=YES,BAND=3707,          X
                UFTI4=(63,22),UFTI5=(51,212)                                @000.000
*       RAMFIL TYPE=4SA,RECNO=00900,RECID=#FILSY,DUPE=YES,BAND=3708,          X
                UFTI4=(63,23),UFTI5=(51,213),EQU=358                      @000.000
*       RAMFIL TYPE=4SA,RECNO=00014,RECID=SPARE,DUPE=YES                      @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(52,250),UFTI5=(62,250),                                X
                USER=(HPN,F,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(52,251),UFTI5=(62,251),                                X
                USER=(HPN,G,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(52,252),UFTI5=(62,252),                                X
                USER=(HPN2,F,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(52,253),UFTI5=(62,253),                                X
                USER=(HPN2,G,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(52,449),UFTI5=(62,449),                                X
                USER=(HPN,H,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(53,220),UFTI5=(63,220),BAND=3320,                    X
                USER=(HPN,B,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(53,221),UFTI5=(63,221),BAND=3321,                    X
                USER=(HPN,C,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X
                UFTI4=(53,222),UFTI5=(63,222),BAND=3322,                    X
                USER=(HPN,D,*)                                              @000.000
*       RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,              X

```

UFTI4=(53,223),UFTI5=(63,223),BAND=3323,	X
USER=(HPN,E,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,224),UFTI5=(63,224),BAND=3324,	X
USER=(HPN,Z,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,225),UFTI5=(63,225),BAND=3325,	X
USER=(HPN,0,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,226),UFTI5=(63,226),BAND=3326,	X
USER=(HPN2,B,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,227),UFTI5=(63,227),BAND=3327,	X
USER=(HPN2,C,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,228),UFTI5=(63,228),BAND=3328,	X
USER=(HPN2,D,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,229),UFTI5=(63,229),BAND=3344,	X
USER=(HPN2,E,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,236),UFTI5=(63,236),BAND=3330,	X
USER=(HPN2,Z,*)	@000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450,	X
UFTI4=(53,237),UFTI5=(63,237),BAND=3331,	X
USER=(HPN2,0,*)	@000.000
RAMFIL TYPE=4SA,RECNO=03000,RECID=#XPRG1,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=03000,RECID=#XPRG2,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=03000,RECID=#XPRG3,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=03000,RECID=#XPRG4,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=00096,RECID=#OSIT,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=01415,RECID=#SRM41A8,DUPE=YES,EQU=301,PRIOR=2	
RAMFIL TYPE=4SA,RECNO=00119,RECID=SPARE,DUPE=YES	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,685),UFTI5=(62,685),USER=(*,L,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,686),UFTI5=(62,686),USER=(*,M,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,687),UFTI5=(62,687),USER=(*,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,688),UFTI5=(62,688),USER=(*,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,689),UFTI5=(62,689),USER=(*,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,690),UFTI5=(62,690),USER=(*,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,691),UFTI5=(62,691),USER=(*,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,692),UFTI5=(62,692),USER=(*,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,693),UFTI5=(62,693),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,694),UFTI5=(62,694),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,695),UFTI5=(62,695),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,696),UFTI5=(62,696),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,889),UFTI5=(62,889),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,890),UFTI5=(62,890),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,891),UFTI5=(62,891),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,892),UFTI5=(62,892),USER=(*,2,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453,	X
UFTI4=(52,893),UFTI5=(62,893),USER=(*,3,*)	@000.000



RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453, UFTI4=(52,894),UFTI5=(62,894),USER=(*,4,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453, UFTI4=(52,895),UFTI5=(62,895),USER=(*,5,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453, UFTI4=(52,896),UFTI5=(62,896),USER=(*,6,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES, UFTI4=(62,78),UFTI5=(52,78),EQU=444,USER=(*,J,*)	X
RAMFIL TYPE=4SA,RECNO=00537,RECID=#XPRG1,DUPE=YES,PRIOR=4	0000.000
RAMFIL TYPE=4SA,RECNO=00537,RECID=#XPRG2,DUPE=YES,PRIOR=4	0000.000
RAMFIL TYPE=4SA,RECNO=00537,RECID=#XPRG3,DUPE=YES,PRIOR=4	0000.000
RAMFIL TYPE=4SA,RECNO=00537,RECID=#XPRG4,DUPE=YES,PRIOR=4	0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#IMQCK,DUPE=YES,EQU=453, UFTI4=(52,448),UFTI5=(62,448),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,450),UFTI5=(62,450), USER=(HPN,I,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,451),UFTI5=(62,451), USER=(HPN,J,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,452),UFTI5=(62,452), USER=(HPN,K,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,453),UFTI5=(62,453), USER=(HPN2,H,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,454),UFTI5=(62,454), USER=(HPN2,I,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,455),UFTI5=(62,455), USER=(HPN2,J,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=18,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,456),UFTI5=(62,456), USER=(HPN2,K,*)	X X 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,649),UFTI5=(62,649),USER=(HPN,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,650),UFTI5=(62,650),USER=(HPN2,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,651),UFTI5=(62,651),USER=(HPN,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00018,RECID=#TPLBL,DUPE=YES,EQU=450, UFTI4=(52,652),UFTI5=(62,652),USER=(HPN2,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES, UFTI4=(62,79),UFTI5=(52,79),EQU=444,USER=(*,K,*)	X X
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,721),UFTI5=(52,721),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,722),UFTI5=(52,722),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,723),UFTI5=(52,723),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,724),UFTI5=(52,724),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,725),UFTI5=(52,725),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,726),UFTI5=(52,726),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,727),UFTI5=(52,727),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,728),UFTI5=(52,728),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,729),UFTI5=(52,729),USER=(*,T,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,730),UFTI5=(52,730),USER=(*,U,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444, UFTI4=(62,731),UFTI5=(52,731),USER=(*,V,*)	X 0000.000

```

RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,732),UFTI5=(52,732),USER=(*,W,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,913),UFTI5=(52,913),USER=(*,X,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,914),UFTI5=(52,914),USER=(*,Y,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,915),UFTI5=(52,915),USER=(*,1,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,916),UFTI5=(52,916),USER=(*,2,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,917),UFTI5=(52,917),USER=(*,3,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,918),UFTI5=(52,918),USER=(*,4,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,919),UFTI5=(52,919),USER=(*,5,*)      @000.000
RAMFIL TYPE=4SA,RECNO=00010,RECID=#Q00J0,DUPE=YES,EQU=444,      X
      UFTI4=(62,920),UFTI5=(52,920),USER=(*,6,*)      @000.000
RAMFIL TYPE=4SA,RECNO=000016,RECID=SPARE,DUPE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 111000
*
* SMALL SHORT TERM POOL RECORDS (SST-A)
* 172 DIRECTORIES OF 770 ADDRESSES EACH
* 132440 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111000,PSON=191400      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111007,PSON=192170      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111014,PSON=192940      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111106,PSON=193710      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111113,PSON=194480      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111205,PSON=195250      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111212,PSON=196020      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111304,PSON=196790      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111311,PSON=197560      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111403,PSON=198330      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111410,PSON=199100      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111502,PSON=199870      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111509,PSON=200640      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111601,PSON=201410      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111608,PSON=202180      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111700,PSON=202950      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111707,PSON=203720      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111714,PSON=204490      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=111806,PSON=205260      @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,      X

```



BASE=111813,PSON=206030	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=111905,PSON=206800	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=111912,PSON=207570	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112004,PSON=208340	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112011,PSON=209110	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112103,PSON=209880	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112110,PSON=210650	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112202,PSON=211420	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112209,PSON=212190	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112301,PSON=212960	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112308,PSON=213730	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112400,PSON=214500	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112407,PSON=215270	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112414,PSON=216040	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112506,PSON=216810	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112513,PSON=217580	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112605,PSON=218350	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112612,PSON=219120	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112704,PSON=219890	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112711,PSON=220660	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112803,PSON=221430	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112810,PSON=222200	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112902,PSON=222970	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=112909,PSON=223740	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113001,PSON=224510	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113008,PSON=225280	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113100,PSON=226050	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113107,PSON=226820	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113114,PSON=227590	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113206,PSON=228360	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113213,PSON=229130	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113305,PSON=229900	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113312,PSON=230670	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=113404,PSON=231440	@000.000

RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113411,PSON=232210	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113503,PSON=232980	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113510,PSON=233750	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113602,PSON=234520	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113609,PSON=235290	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113701,PSON=236060	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113708,PSON=236830	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113800,PSON=237600	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113807,PSON=238370	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113814,PSON=239140	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113906,PSON=239910	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=113913,PSON=240680	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114005,PSON=241450	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114012,PSON=242220	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114104,PSON=242990	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114111,PSON=243760	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114203,PSON=244530	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114210,PSON=245300	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114302,PSON=246070	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114309,PSON=246840	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114401,PSON=247610	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114408,PSON=248380	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114500,PSON=249150	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114507,PSON=249920	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114514,PSON=250690	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114606,PSON=251460	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114613,PSON=252230	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114705,PSON=253000	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114712,PSON=253770	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114804,PSON=254540	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114811,PSON=255310	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114903,PSON=256080	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=114910,PSON=256850	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X

BASE=115002,PSON=257620	@000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115009,PSON=258390	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115101,PSON=259160	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115108,PSON=259930	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115200,PSON=260700	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115207,PSON=261470	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115214,PSON=262240	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115306,PSON=263010	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115313,PSON=263780	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115405,PSON=264550	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115412,PSON=265320	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115504,PSON=266090	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115511,PSON=266860	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115603,PSON=267630	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115610,PSON=268400	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115702,PSON=269170	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115709,PSON=269940	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115801,PSON=270710	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115808,PSON=271480	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115900,PSON=272250	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115907,PSON=273020	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=115914,PSON=273790	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116006,PSON=274560	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116013,PSON=275330	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116105,PSON=276100	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116112,PSON=276870	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116204,PSON=277640	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116211,PSON=278410	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116303,PSON=279180	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116310,PSON=279950	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116402,PSON=280720	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116409,PSON=281490	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116501,PSON=282260	X @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116508,PSON=283030	X @000.000

RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116600,PSON=283800	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116607,PSON=284570	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116614,PSON=285340	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116706,PSON=286110	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116713,PSON=286880	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116805,PSON=287650	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116812,PSON=288420	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116904,PSON=289190	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=116911,PSON=289960	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117003,PSON=290730	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117010,PSON=291500	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117102,PSON=292270	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117109,PSON=293040	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117201,PSON=293810	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117208,PSON=294580	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117300,PSON=295350	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117307,PSON=296120	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117314,PSON=296890	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117406,PSON=297660	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117413,PSON=298430	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117505,PSON=299200	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117512,PSON=299970	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117604,PSON=300740	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117611,PSON=301510	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117703,PSON=302280	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117710,PSON=303050	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117802,PSON=303820	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117809,PSON=304590	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117901,PSON=305360	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=117908,PSON=306130	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=118000,PSON=306900	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=118007,PSON=307670	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, BASE=118014,PSON=308440	X 0000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST,	X

```

BASE=118106,PSON=309210                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118113,PSON=309980                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118205,PSON=310750                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118212,PSON=311520                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118304,PSON=312290                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118311,PSON=313060                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118403,PSON=313830                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118410,PSON=314600                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118502,PSON=315370                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118509,PSON=316140                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118601,PSON=316910                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118608,PSON=317680                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118700,PSON=318450                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118707,PSON=319220                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118714,PSON=319990                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118806,PSON=320760                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118813,PSON=321530                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118905,PSON=322300                                @000.000
RAMFIL TYPE=SSA,RECNO=00770,RECID=POOL,DUPE=NO,POLID=ST, X
BASE=118912,PSON=323070                                @000.000
*
*****
*
* NEXT BASE ADDRESS IS 119004
*
* SMALL LONG TERM DUPED POOL RECORDS (SDP-A)
* 73 DIRECTORIES OF 1100 ADDRESSES EACH
* 80300 RECORDS TOTAL
*
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119004,PSON=145200                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119014,PSON=146300                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119109,PSON=147400                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119204,PSON=148500                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119214,PSON=149600                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119309,PSON=150700                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119404,PSON=151800                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119414,PSON=152900                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119509,PSON=154000                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=119604,PSON=155100                                @000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, X

```

BASE=119614,PSON=156200	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=119709,PSON=157300	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=119804,PSON=158400	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=119814,PSON=159500	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=119909,PSON=160600	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120004,PSON=161700	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120014,PSON=162800	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120109,PSON=163900	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120204,PSON=165000	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120214,PSON=166100	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120309,PSON=167200	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120404,PSON=168300	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120414,PSON=169400	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120509,PSON=170500	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120604,PSON=171600	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120614,PSON=172700	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120709,PSON=173800	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120804,PSON=174900	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120814,PSON=176000	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=120909,PSON=177100	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121004,PSON=178200	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121014,PSON=179300	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121109,PSON=180400	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121204,PSON=181500	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121214,PSON=182600	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121309,PSON=183700	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121404,PSON=184800	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121414,PSON=185900	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121509,PSON=187000	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121604,PSON=188100	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121614,PSON=189200	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121709,PSON=190300	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121804,PSON=191400	@000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=121814,PSON=192500	@000.000



RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=121909,PSON=193600	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122004,PSON=194700	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122014,PSON=195800	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122109,PSON=196900	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122204,PSON=198000	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122214,PSON=199100	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122309,PSON=200200	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122404,PSON=201300	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122414,PSON=202400	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122509,PSON=203500	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122604,PSON=204600	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122614,PSON=205700	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122709,PSON=206800	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122804,PSON=207900	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122814,PSON=209000	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=122909,PSON=210100	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123004,PSON=211200	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123014,PSON=212300	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123109,PSON=213400	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123204,PSON=214500	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123214,PSON=215600	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123309,PSON=216700	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123404,PSON=217800	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123414,PSON=218900	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123509,PSON=220000	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123604,PSON=221100	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123614,PSON=222200	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123709,PSON=223300	X 0000.000
RAMFIL TYPE=SSA,RECNO=01100,RECID=POOL,DUPE=YES,POLID=LT, BASE=123804,PSON=224400	X 0000.000
* *****	
* * NEXT BASE ADDRESS IS 123814 * * SMALL DUPED FIXED FILE RECORDS * 28160 RECORDS TOTAL *	
RAMFIL TYPE=SSA,RECNO=028160,RECID=SPARE,DUPE=YES,BASE=123814	0000.000

```

*
*****
*
* NEXT BASE ADDRESS IS 125600
*
* LARGE SHORT TERM POOL RECORDS (LST-A)
* 49 DIRECTORIES OF 462 ADDRESSES EACH
* PSON GAP: ORDINALS 69300-74999 ARE UNDEFINED
* 22638 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125600,PSON=75000      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125607,PSON=75462      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125614,PSON=75924      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125706,PSON=76386      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125713,PSON=76848      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125805,PSON=77310      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125812,PSON=77772      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125904,PSON=78234      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=125911,PSON=78696      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126003,PSON=79158      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126010,PSON=79620      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126102,PSON=80082      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126109,PSON=80544      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126201,PSON=81006      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126208,PSON=81468      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126300,PSON=81930      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126307,PSON=82392      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126314,PSON=82854      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126406,PSON=83316      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126413,PSON=83778      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126505,PSON=84240      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126512,PSON=84702      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126604,PSON=85164      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126611,PSON=85626      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126703,PSON=86088      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126710,PSON=86550      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126802,PSON=87012      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X
      BASE=126809,PSON=87474      @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,          X

```



```

      BASE=126901,PSON=87936                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=126908,PSON=88398                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127000,PSON=88860                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127007,PSON=89322                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127014,PSON=89784                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127106,PSON=90246                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127113,PSON=90708                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127205,PSON=91170                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127212,PSON=91632                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127304,PSON=92094                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127311,PSON=92556                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127403,PSON=93018                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127410,PSON=93480                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127502,PSON=93942                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127509,PSON=94404                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127601,PSON=94866                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127608,PSON=95328                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127700,PSON=95790                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127707,PSON=96252                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127714,PSON=96714                                @000.000
RAMFIL TYPE=LSA,RECNO=00462,RECID=POOL,DUPE=NO,POLID=ST,    X
      BASE=127806,PSON=97176                                @000.000
*
*****
*
* NEXT BASE ADDRESS IS 127813
*
* LARGE LONG TERM DUPED POOL RECORDS (LDP-A)
* 35 DIRECTORIES OF 990 ADDRESSES EACH
* 35 DIRECTORIES OF 1650 ADDRESSES EACH
* 111 DIRECTORIES OF 1320 ADDRESSES EACH
* PSON GAPS: ORDINALS 310570 - 319999 ARE UNDEFINED
* BEFORE EACH DIRECTORY WITH 1650 ADDRESSES,
* THE PREVIOUS 1100 ORDINALS ARE UNDEFINED
* 238920 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=127813,PSON=136620                                @000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=128003,PSON=137940                                @000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=128103,PSON=139940                                @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=128213,PSON=141590                                @000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=128403,PSON=142910                                @000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=128503,PSON=144910                                @000.000

```

RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=128613,PSON=146560	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=128803,PSON=147880	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=128903,PSON=149880	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=129013,PSON=151530	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=129203,PSON=152850	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=129303,PSON=154850	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=129413,PSON=156500	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=129603,PSON=157820	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=129703,PSON=159820	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=129813,PSON=161470	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=130003,PSON=162790	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=130103,PSON=164790	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=130213,PSON=166440	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=130403,PSON=167760	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=130503,PSON=169760	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=130613,PSON=171410	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=130803,PSON=172730	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=130903,PSON=174730	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=131013,PSON=176380	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=131203,PSON=177700	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=131303,PSON=179700	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=131413,PSON=181350	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=131603,PSON=182670	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=131703,PSON=184670	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=131813,PSON=186320	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=132003,PSON=187640	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=132103,PSON=189640	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=132213,PSON=191290	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=132403,PSON=192610	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=132503,PSON=194610	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=132613,PSON=196260	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=132803,PSON=197580	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=132903,PSON=199580	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X

BASE=133013,PSON=201230	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133203,PSON=202550	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133303,PSON=204550	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133413,PSON=206200	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133603,PSON=207520	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133703,PSON=209520	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=133813,PSON=211170	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134003,PSON=212490	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134103,PSON=214490	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134213,PSON=216140	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134403,PSON=217460	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134503,PSON=219460	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134613,PSON=221110	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134803,PSON=222430	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=134903,PSON=224430	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135013,PSON=226080	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135203,PSON=227400	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135303,PSON=229400	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135413,PSON=231050	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135603,PSON=232370	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135703,PSON=234370	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=135813,PSON=236020	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136003,PSON=237340	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136103,PSON=239340	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136213,PSON=240990	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136403,PSON=242310	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136503,PSON=244310	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136613,PSON=245960	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136803,PSON=247280	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=136903,PSON=249280	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=137013,PSON=250930	@000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=137203,PSON=252250	@000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=137303,PSON=254250	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=137413,PSON=255900	@000.000

RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=137603,PSON=257220	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=137703,PSON=259220	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=137813,PSON=260870	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=138003,PSON=262190	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=138103,PSON=264190	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=138213,PSON=265840	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=138403,PSON=267160	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=138503,PSON=269160	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=138613,PSON=270810	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=138803,PSON=272130	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=138903,PSON=274130	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=139013,PSON=275780	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=139203,PSON=277100	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=139303,PSON=279100	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=139413,PSON=280750	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=139603,PSON=282070	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=139703,PSON=284070	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=139813,PSON=285720	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=140003,PSON=287040	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=140103,PSON=289040	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=140213,PSON=290690	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=140403,PSON=292010	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=140503,PSON=294010	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=140613,PSON=295660	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=140803,PSON=296980	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=140903,PSON=298980	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=141013,PSON=300630	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=141203,PSON=301950	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=141303,PSON=303950	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=141413,PSON=305600	X 0000.000
RAMFIL TYPE=LSA,RECNO=00990,RECID=POOL,DUPE=YES,POLID=LT, BASE=141603,PSON=306920	X 0000.000
RAMFIL TYPE=LSA,RECNO=01650,RECID=POOL,DUPE=YES,POLID=LT, BASE=141703,PSON=308920	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=141813,PSON=320000	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X

BASE=142003,PSON=321320	@000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142108,PSON=322640	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142213,PSON=323960	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142403,PSON=325280	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142508,PSON=326600	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142613,PSON=327920	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142803,PSON=329240	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=142908,PSON=330560	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143013,PSON=331880	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143203,PSON=333200	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143308,PSON=334520	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143413,PSON=335840	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143603,PSON=337160	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143708,PSON=338480	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=143813,PSON=339800	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144003,PSON=341120	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144108,PSON=342440	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144213,PSON=343760	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144403,PSON=345080	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144508,PSON=346400	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144613,PSON=347720	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144803,PSON=349040	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=144908,PSON=350360	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145013,PSON=351680	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145203,PSON=353000	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145308,PSON=354320	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145413,PSON=355640	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145603,PSON=356960	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145708,PSON=358280	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=145813,PSON=359600	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146003,PSON=360920	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146108,PSON=362240	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146213,PSON=363560	X @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146403,PSON=364880	X @000.000

RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146508,PSON=366200	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146613,PSON=367520	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146803,PSON=368840	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=146908,PSON=370160	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147013,PSON=371480	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147203,PSON=372800	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147308,PSON=374120	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147413,PSON=375440	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147603,PSON=376760	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147708,PSON=378080	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=147813,PSON=379400	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148003,PSON=380720	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148108,PSON=382040	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148213,PSON=383360	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148403,PSON=384680	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148508,PSON=386000	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148613,PSON=387320	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148803,PSON=388640	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=148908,PSON=389960	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149013,PSON=391280	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149203,PSON=392600	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149308,PSON=393920	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149413,PSON=395240	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149603,PSON=396560	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149708,PSON=397880	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=149813,PSON=399200	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150003,PSON=400520	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150108,PSON=401840	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150213,PSON=403160	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150403,PSON=404480	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150508,PSON=405800	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150613,PSON=407120	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, BASE=150803,PSON=408440	X 0000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT,	X



```

BASE=150908,PSON=409760 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151013,PSON=411080 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151203,PSON=412400 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151308,PSON=413720 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151413,PSON=415040 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151603,PSON=416360 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151708,PSON=417680 @000.000
RAMFIL TYPE=LSA,RECNO=01320,RECID=POOL,DUPE=YES,POLID=LT, X
BASE=151813,PSON=419000 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 152003
*
* LARGE DUPED FIXED FILE RECORDS
* 52536 RECORDS TOTAL
*
RAMFIL TYPE=LSA,RECNO=01050,RECID=#SONRI,DUPE=YES,EQU=89, X
BASE=152003,PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=01050,RECID=#STPKP,DUPE=YES,EQU=91,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONCP,DUPE=YES,EQU=370,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONUP,DUPE=YES,EQU=371,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONSV,DUPE=YES,EQU=372,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONDE,DUPE=YES,EQU=373,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONROLL,DUPE=YES,EQU=378,PRIOR=2
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPM,DUPE=YES,EQU=379,PRIOR=2
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE0,DUPE=YES,EQU=290, X
UFTI4=(62,11),UFTI5=(52,11) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE1,DUPE=YES,EQU=291, X
UFTI4=(62,12),UFTI5=(52,12) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE2,DUPE=YES,EQU=292, X
UFTI4=(62,14),UFTI5=(52,14) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE3,DUPE=YES,EQU=293, X
UFTI4=(62,16),UFTI5=(52,16) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE4,DUPE=YES,EQU=294, X
UFTI4=(62,17),UFTI5=(52,17) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE5,DUPE=YES,EQU=295, X
UFTI4=(62,18),UFTI5=(52,18) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE6,DUPE=YES,EQU=296, X
UFTI4=(62,19),UFTI5=(52,19) @000.000
RAMFIL TYPE=LSA,RECNO=001300,RECID=#SONRPE7,DUPE=YES,EQU=297, X
UFTI4=(62,20),UFTI5=(52,20) @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,B,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,C,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,D,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,E,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,Z,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00975,RECID=#SONRPE,DUPE=YES,EQU=377, X
USER=(*,0,*),PRIOR=2 @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333, X
UFTI4=(61,1014),UFTI5=(51,1014),USER=(*,J,*) @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333, X
UFTI4=(61,497),UFTI5=(51,497),USER=(*,K,*) @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333, X
UFTI4=(61,601),UFTI5=(51,601),USER=(*,L,*) @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333, X

```

```

      UFTI4=(61,602),UFTI5=(51,602),USER=(*,M,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,603),UFTI5=(51,603),USER=(*,N,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,604),UFTI5=(51,604),USER=(*,O,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,605),UFTI5=(51,605),USER=(*,P,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,606),UFTI5=(51,606),USER=(*,Q,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,607),UFTI5=(51,607),USER=(*,R,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,608),UFTI5=(51,608),USER=(*,S,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,609),UFTI5=(51,609),USER=(*,T,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,610),UFTI5=(51,610),USER=(*,U,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,611),UFTI5=(51,611),USER=(*,V,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,612),UFTI5=(51,612),USER=(*,W,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,830),UFTI5=(51,830),USER=(*,X,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,831),UFTI5=(51,831),USER=(*,Y,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,832),UFTI5=(51,832),USER=(*,1,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,833),UFTI5=(51,833),USER=(*,2,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,834),UFTI5=(51,834),USER=(*,3,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,835),UFTI5=(51,835),USER=(*,4,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,836),UFTI5=(51,836),USER=(*,5,*)      @000.000
RAMFIL TYPE=LSA,RECNO=00097,RECID=#SONSKP,DUPE=YES,EQU=333,      X
      UFTI4=(61,837),UFTI5=(51,837),USER=(*,6,*)      @000.000
RAMFIL TYPE=LSA,RECNO=026202,RECID=SPARE,DUPE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 157304
*
* 4K LONG TERM DUPED POOL RECORDS (4DP-A)
* 62 DIRECTORIES OF 1440 ADDRESSES EACH
* 2 DIRECTORIES OF 1456 ADDRESSES EACH
* 5 DIRECTORIES OF 8000 ADDRESSES EACH
* PS0N GAP:  ORDINALS 148776 - 199999 ARE UNDEFINED
* 132192 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=157304,PS0N=124920      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=157704,PS0N=126360      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=158104,PS0N=127800      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=158504,PS0N=129240      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=158904,PS0N=130680      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=159304,PS0N=132120      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=159704,PS0N=133560      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=160104,PS0N=135000      @000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X

```



BASE=160504,PSON=136440	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=160904,PSON=137880	@000.000
RAMFIL TYPE=4SA,RECNO=09456,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=161304,PSON=139320	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=163908,PSON=208592	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=164308,PSON=210032	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=164708,PSON=211472	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=165108,PSON=212912	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=165508,PSON=214352	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=165908,PSON=215792	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=166308,PSON=217232	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=166708,PSON=218672	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=167108,PSON=220112	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=167508,PSON=221552	@000.000
RAMFIL TYPE=4SA,RECNO=09456,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=167908,PSON=222992	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=170512,PSON=232448	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=170912,PSON=233888	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=171312,PSON=235328	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=171712,PSON=236768	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=172112,PSON=238208	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=172512,PSON=239648	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=172912,PSON=241088	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=173312,PSON=242528	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=173712,PSON=243968	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=174112,PSON=245408	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=174512,PSON=246848	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=174912,PSON=248288	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=175312,PSON=249728	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=175712,PSON=251168	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=176112,PSON=252608	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=176512,PSON=254048	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=176912,PSON=255488	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=177312,PSON=256928	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=177712,PSON=258368	@000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X
BASE=178112,PSON=259808	@000.000

RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=178512,PSON=261248	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=178912,PSON=262688	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=179312,PSON=264128	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=179712,PSON=265568	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=180112,PSON=267008	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=180512,PSON=268448	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=180912,PSON=269888	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=181312,PSON=271328	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=181712,PSON=272768	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=182112,PSON=274208	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=182512,PSON=275648	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=182912,PSON=277088	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=183312,PSON=278528	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=183712,PSON=279968	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=184112,PSON=281408	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=184512,PSON=282848	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=184912,PSON=284288	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=185312,PSON=285728	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=185712,PSON=287168	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=186112,PSON=288608	X 0000.000
RAMFIL TYPE=4SA,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=186512,PSON=290048	X 0000.000
RAMFIL TYPE=4SA,RECNO=25440,RECID=POOL,DUPE=YES,POLID=LT, BASE=186912,PSON=291488	X 0000.000
*	
*****	
*	
* NEXT BASE ADDRESS IS 194007	
*	
* 4K SHORT TERM POOL RECORDS (4ST-A)	
* 54 DIRECTORIES OF 480 ADDRESSES EACH	
* 1 DIRECTORY OF 488 ADDRESSES	
* 5 DIRECTORIES OF 8000 ADDRESSES EACH	
* PSON GAP: ORDINALS 274224 - 299999 ARE UNDEFINED	
* 66408 RECORDS TOTAL	
*	
RAMFIL TYPE=4SA,RECNO=16488,RECID=POOL,DUPE=NO,POLID=ST, BASE=194007,PSON=300000	X 0000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST, BASE=198604,PSON=316488	X 0000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST, BASE=198709,PSON=316968	X 0000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST, BASE=198814,PSON=317448	X 0000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST, BASE=199004,PSON=317928	X 0000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X

BASE=199109,PSON=318408	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199214,PSON=318888	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199404,PSON=319368	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199509,PSON=319848	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199614,PSON=320328	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199804,PSON=320808	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=199909,PSON=321288	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200014,PSON=321768	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200204,PSON=322248	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200309,PSON=322728	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200414,PSON=323208	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200604,PSON=323688	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200709,PSON=324168	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=200814,PSON=324648	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201004,PSON=325128	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201109,PSON=325608	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201214,PSON=326088	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201404,PSON=326568	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201509,PSON=327048	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201614,PSON=327528	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201804,PSON=328008	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=201909,PSON=328488	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202014,PSON=328968	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202204,PSON=329448	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202309,PSON=329928	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202414,PSON=330408	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202604,PSON=330888	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202709,PSON=331368	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=202814,PSON=331848	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=203004,PSON=332328	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=203109,PSON=332808	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=203214,PSON=333288	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=203404,PSON=333768	@000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=203509,PSON=334248	@000.000

```

RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=203614,PSON=334728      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=203804,PSON=335208      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=203909,PSON=335688      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204014,PSON=336168      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204204,PSON=336648      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204309,PSON=337128      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204414,PSON=337608      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204604,PSON=338088      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204709,PSON=338568      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=204814,PSON=339048      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205004,PSON=339528      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205109,PSON=340008      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205214,PSON=340488      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205404,PSON=340968      @000.000
RAMFIL TYPE=4SA,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205509,PSON=341448      @000.000
RAMFIL TYPE=4SA,RECNO=24480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=205614,PSON=341928      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 212414
*
* 4K DUPED FIXED FILE RECORDS (VERTICALLY ALLOCATED)
* 2400 RECORDS TOTAL
*
* Note that some #KFBXnnn records are purposely split to
* test the code that handles split keypoint extents.
*
RAMFIL TYPE=4SA,RECNO=000240,RECID=#KFBX0,DUPE=YES,EQU=268,      X
      UFTI4=(61,226),UFTI5=(51,226),BASE=212414      @000.000
RAMFIL TYPE=4SA,RECNO=000240,RECID=#KFBX1,DUPE=YES,EQU=269,      X
      UFTI4=(61,227),UFTI5=(51,227),BASE=212604      @000.000
RAMFIL TYPE=4SA,RECNO=000060,RECID=#KFBX2,DUPE=YES,EQU=311,      X
      UFTI4=(52,100),UFTI5=(62,100),BASE=212709,PRIOR=1
RAMFIL TYPE=4SA,RECNO=000168,RECID=#KFBX3,DUPE=YES,EQU=312,      X
      UFTI4=(52,101),UFTI5=(62,101),BASE=212714,PRIOR=1
RAMFIL TYPE=4SA,RECNO=000120,RECID=#KFBX2,DUPE=YES,      X
      BASE=212813,PRIOR=2      @000.000
RAMFIL TYPE=4SA,RECNO=000072,RECID=#KFBX3,DUPE=YES,      X
      BASE=212908,PRIOR=2      @000.000
RAMFIL TYPE=4SA,RECNO=000060,RECID=#KFBX2,DUPE=YES,      X
      BASE=212914,PRIOR=3      @000.000
RAMFIL TYPE=4SA,RECNO=000240,RECID=#KFBX4,DUPE=YES,EQU=313,      X
      UFTI4=(52,102),UFTI5=(62,102),BASE=213004      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 213109
*
* 4K DUPED FIXED FILE RECORDS
* 31296 RECORDS TOTAL
*

```

```

RAMFIL TYPE=4SA,RECNO=002150,RECID=SPARE,DUPE=YES,BASE=213109 @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,258),UFTI5=(62,258),USER=(*,F,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,259),UFTI5=(62,259),USER=(*,G,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,518),UFTI5=(62,518),USER=(*,H,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,519),UFTI5=(62,519),USER=(*,I,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,520),UFTI5=(62,520),USER=(*,J,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,521),UFTI5=(62,521),USER=(*,K,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,673),UFTI5=(62,673),USER=(*,L,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,674),UFTI5=(62,674),USER=(*,M,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,675),UFTI5=(62,675),USER=(*,N,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,676),UFTI5=(62,676),USER=(*,O,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,677),UFTI5=(62,677),USER=(*,P,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,678),UFTI5=(62,678),USER=(*,Q,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,679),UFTI5=(62,679),USER=(*,R,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,680),UFTI5=(62,680),USER=(*,S,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,681),UFTI5=(62,681),USER=(*,T,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,682),UFTI5=(62,682),USER=(*,U,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,683),UFTI5=(62,683),USER=(*,V,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,684),UFTI5=(62,684),USER=(*,W,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,881),UFTI5=(62,881),USER=(*,X,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,882),UFTI5=(62,882),USER=(*,Y,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,883),UFTI5=(62,883),USER=(*,1,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,884),UFTI5=(62,884),USER=(*,2,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,885),UFTI5=(62,885),USER=(*,3,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,886),UFTI5=(62,886),USER=(*,4,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,887),UFTI5=(62,887),USER=(*,5,*) @000.000
RAMFIL TYPE=4SA,RECNO=00100,RECID=#IBMMP4,DUPE=YES,EQU=452, X
      UFTI4=(52,888),UFTI5=(62,888),USER=(*,6,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,974),UFTI5=(51,974),USER=(*,F,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,975),UFTI5=(51,975),USER=(*,G,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,558),UFTI5=(51,558),USER=(*,H,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,559),UFTI5=(51,559),USER=(*,I,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,560),UFTI5=(51,560),USER=(*,J,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,561),UFTI5=(51,561),USER=(*,K,*) @000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, X
      UFTI4=(61,781),UFTI5=(51,781),USER=(*,L,*) @000.000

```



RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,782),UFTI5=(51,782),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,783),UFTI5=(51,783),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,784),UFTI5=(51,784),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,785),UFTI5=(51,785),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,786),UFTI5=(51,786),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,787),UFTI5=(51,787),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,788),UFTI5=(51,788),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,789),UFTI5=(51,789),USER=(*,T,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,790),UFTI5=(51,790),USER=(*,U,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,791),UFTI5=(51,791),USER=(*,V,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,792),UFTI5=(51,792),USER=(*,W,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,950),UFTI5=(51,950),USER=(*,X,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,951),UFTI5=(51,951),USER=(*,Y,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,952),UFTI5=(51,952),USER=(*,1,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,953),UFTI5=(51,953),USER=(*,2,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,954),UFTI5=(51,954),USER=(*,3,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,955),UFTI5=(51,955),USER=(*,4,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,956),UFTI5=(51,956),USER=(*,5,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00020,RECID=#STPUR,DUPE=YES,EQU=256, UFTI4=(61,957),UFTI5=(51,957),USER=(*,6,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,293),UFTI5=(62,293),USER=(*,F,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,294),UFTI5=(62,294),USER=(*,G,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,431),UFTI5=(62,431),USER=(*,H,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,432),UFTI5=(62,432),USER=(*,I,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,433),UFTI5=(62,433),USER=(*,J,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=30,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,434),UFTI5=(62,434),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,709),UFTI5=(62,709),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,710),UFTI5=(62,710),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,711),UFTI5=(62,711),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,712),UFTI5=(62,712),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,713),UFTI5=(62,713),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,714),UFTI5=(62,714),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,715),UFTI5=(62,715),USER=(*,R,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451, UFTI4=(52,716),UFTI5=(62,716),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X

UFTI4=(52,717),UFTI5=(62,717),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,718),UFTI5=(62,718),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,719),UFTI5=(62,719),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,720),UFTI5=(62,720),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,905),UFTI5=(62,905),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,906),UFTI5=(62,906),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,907),UFTI5=(62,907),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,908),UFTI5=(62,908),USER=(*,2,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,909),UFTI5=(62,909),USER=(*,3,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,910),UFTI5=(62,910),USER=(*,4,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,911),UFTI5=(62,911),USER=(*,5,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00030,RECID=#TDTDR,DUPE=YES,EQU=451,	X
UFTI4=(52,912),UFTI5=(62,912),USER=(*,6,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,996),UFTI5=(51,996),USER=(*,F,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,997),UFTI5=(51,997),USER=(*,G,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,530),UFTI5=(51,530),USER=(*,H,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,531),UFTI5=(51,531),USER=(*,I,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,532),UFTI5=(51,532),USER=(*,J,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,533),UFTI5=(51,533),USER=(*,K,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,697),UFTI5=(51,697),USER=(*,L,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,698),UFTI5=(51,698),USER=(*,M,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,699),UFTI5=(51,699),USER=(*,N,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,700),UFTI5=(51,700),USER=(*,O,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,701),UFTI5=(51,701),USER=(*,P,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,702),UFTI5=(51,702),USER=(*,Q,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,703),UFTI5=(51,703),USER=(*,R,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,704),UFTI5=(51,704),USER=(*,S,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,705),UFTI5=(51,705),USER=(*,T,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,706),UFTI5=(51,706),USER=(*,U,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,707),UFTI5=(51,707),USER=(*,V,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,708),UFTI5=(51,708),USER=(*,W,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,894),UFTI5=(51,894),USER=(*,X,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,895),UFTI5=(51,895),USER=(*,Y,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,896),UFTI5=(51,896),USER=(*,1,*)	@000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161,	X
UFTI4=(61,897),UFTI5=(51,897),USER=(*,2,*)	@000.000

RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161, UFTI4=(61,898),UFTI5=(51,898),USER=(*,3,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161, UFTI4=(61,899),UFTI5=(51,899),USER=(*,4,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161, UFTI4=(61,900),UFTI5=(51,900),USER=(*,5,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00150,RECID=#RT1RI,DUPE=YES,EQU=161, UFTI4=(61,901),UFTI5=(51,901),USER=(*,6,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,740),UFTI5=(51,740),USER=(*,S,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,741),UFTI5=(51,741),USER=(*,T,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,742),UFTI5=(51,742),USER=(*,U,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,743),UFTI5=(51,743),USER=(*,V,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,744),UFTI5=(51,744),USER=(*,W,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,918),UFTI5=(51,918),USER=(*,X,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,919),UFTI5=(51,919),USER=(*,Y,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,920),UFTI5=(51,920),USER=(*,1,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,921),UFTI5=(51,921),USER=(*,2,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,922),UFTI5=(51,922),USER=(*,3,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,923),UFTI5=(51,923),USER=(*,4,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,924),UFTI5=(51,924),USER=(*,5,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00180,RECID=#RV2RU,DUPE=YES,EQU=467, UFTI4=(61,925),UFTI5=(51,925),USER=(*,6,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=01415,RECID=#SRM51A8,DUPE=YES,EQU=302,PRIOR=2	
RAMFIL TYPE=4SA,RECNO=00119,RECID=SPARE,DUPE=YES	0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,47),UFTI5=(52,47),USER=(*,F,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,48),UFTI5=(52,48),USER=(*,G,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,49),UFTI5=(52,49),USER=(*,H,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,71),UFTI5=(52,71),USER=(*,I,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,72),UFTI5=(52,72),USER=(*,J,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,73),UFTI5=(52,73),USER=(*,K,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,45),UFTI5=(52,45),USER=(*,L,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,46),UFTI5=(52,46),USER=(*,M,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,94),UFTI5=(52,94),USER=(*,N,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,95),UFTI5=(52,95),USER=(*,O,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,96),UFTI5=(52,96),USER=(*,P,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462, UFTI4=(62,97),UFTI5=(52,97),USER=(*,Q,*)	X 0000.000
RAMFIL TYPE=4SA,RECNO=000532,RECID=#SRM41A8,DUPE=YES,BAND=3922, UFTI4=((61,458),(61,459),(62,92)), UFTI5=((51,458),(51,303)),EQU=301,PRIOR=1	X X 0000.000
RAMFIL TYPE=4SA,RECNO=000532,RECID=#SRM51A8,DUPE=YES,BAND=3923, UFTI4=((61,460),(61,461),(62,93)), UFTI5=((51,460),(51,307)),EQU=302,PRIOR=1	X X 0000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BREATB8,DUPE=YES,BAND=3924,	X



```

      UFTI4=(61,462),UFTI5=(51,462),EQU=303          @000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRL0TB8,DUPE=YES,BAND=3925,      X
      UFTI4=(61,463),UFTI5=(51,463),EQU=304          @000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDT08,DUPE=YES,BAND=3926,      X
      UFTI4=(61,464),UFTI5=(51,464),EQU=305,USER=HPN  @000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDT08,DUPE=YES,BAND=3927,      X
      UFTI4=(61,465),UFTI5=(51,465),EQU=305,USER=HPN2 @000.000
RAMFIL TYPE=4SA,RECNO=001290,RECID=#BRHIST8,DUPE=YES,BAND=3928,      X
      UFTI4=((61,466),(61,467),(61,495)),UFTI5=(51,466),      X
      EQU=306,USER=HPN          @000.000
RAMFIL TYPE=4SA,RECNO=001290,RECID=#BRHIST8,DUPE=YES,BAND=3929,      X
      UFTI4=((61,468),(61,469),(61,496)),UFTI5=(51,468),      X
      EQU=306,USER=HPN2          @000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDDE8,DUPE=YES,BAND=3930,      X
      UFTI4=(61,470),UFTI5=(51,470),EQU=307          @000.000
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3931,      X
      UFTI4=(61,471),UFTI5=(51,471),EQU=308,USER=(HPN,B,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3932,      X
      UFTI4=(61,472),UFTI5=(51,472),EQU=308,USER=(HPN,C,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3933,      X
      UFTI4=(61,473),UFTI5=(51,473),EQU=308,USER=(HPN,D,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3934,      X
      UFTI4=(61,474),UFTI5=(51,474),EQU=308,USER=(HPN,E,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3935,      X
      UFTI4=(61,475),UFTI5=(51,475),EQU=308,USER=(HPN,Z,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3936,      X
      UFTI4=(61,476),UFTI5=(51,476),EQU=308,USER=(HPN,0,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3937,      X
      UFTI4=(61,477),UFTI5=(51,477),EQU=308,USER=(HPN2,B,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3938,      X
      UFTI4=(61,478),UFTI5=(51,478),EQU=308,USER=(HPN2,C,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3939,      X
      UFTI4=(61,479),UFTI5=(51,479),EQU=308,USER=(HPN2,D,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3940,      X
      UFTI4=(61,480),UFTI5=(51,480),EQU=308,USER=(HPN2,E,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3941,      X
      UFTI4=(61,481),UFTI5=(51,481),EQU=308,USER=(HPN2,Z,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDSA8,DUPE=YES,BAND=3942,      X
      UFTI4=(61,482),UFTI5=(51,482),EQU=308,USER=(HPN2,0,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3943,      X
      UFTI4=(61,483),UFTI5=(51,483),EQU=309,USER=(HPN,B,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3944,      X
      UFTI4=(61,484),UFTI5=(51,484),EQU=309,USER=(HPN,C,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3945,      X
      UFTI4=(61,485),UFTI5=(51,485),EQU=309,USER=(HPN,D,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3946,      X
      UFTI4=(61,486),UFTI5=(51,486),EQU=309,USER=(HPN,E,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3947,      X
      UFTI4=(61,487),UFTI5=(51,487),EQU=309,USER=(HPN,Z,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3948,      X
      UFTI4=(61,488),UFTI5=(51,488),EQU=309,USER=(HPN,0,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3949,      X
      UFTI4=(61,489),UFTI5=(51,489),EQU=309,USER=(HPN2,B,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3950,      X
      UFTI4=(61,490),UFTI5=(51,490),EQU=309,USER=(HPN2,C,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3951,      X
      UFTI4=(61,491),UFTI5=(51,491),EQU=309,USER=(HPN2,D,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3952,      X
      UFTI4=(61,492),UFTI5=(51,492),EQU=309,USER=(HPN2,E,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3953,      X
      UFTI4=(61,493),UFTI5=(51,493),EQU=309,USER=(HPN2,Z,*)
RAMFIL TYPE=4SA,RECNO=000129,RECID=#BRIDTB8,DUPE=YES,BAND=3954,      X
      UFTI4=(61,494),UFTI5=(51,494),EQU=309,USER=(HPN2,0,*)
RAMFIL TYPE=4SA,RECNO=000007,RECID=SPARE,DUPE=YES          @000.000

```

\*

\*\*\*\*\*

\*

```

* NEXT BASE ADDRESS IS 221808
*
*      4K FARF6 LONG TERM DUPED POOL RECORDS (4D6-A)
*      77 DIRECTORIES OF 360 ADDRESSES EACH
*      8 DIRECTORIES OF 3600 ADDRESSES EACH
*      1 DIRECTORY OF 4800 ADDRESSES
*      4 DIRECTORIES OF 6400 ADDRESSES EACH
*      7 DIRECTORIES OF 8000 ADDRESSES EACH
*      142920 RECORDS TOTAL
*
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=221808,PSON=0,UFTI6=((51,20),(51,30),(51,40))
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=222808,PSON=3600                                @000.000
RAMFIL TYPE=4SA,RECNO=14400,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=223808,PSON=7200                                @000.000
RAMFIL TYPE=4SA,RECNO=14400,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=227808,PSON=21600                               @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=231808,PSON=36000                               @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=232808,PSON=39600                               @000.000
RAMFIL TYPE=4SA,RECNO=14400,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=233808,PSON=43200                               @000.000
RAMFIL TYPE=4SA,RECNO=14400,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=237808,PSON=57600                               @000.000
RAMFIL TYPE=4SA,RECNO=28800,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=241808,PSON=72000                               @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=249808,PSON=100800                              @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=250808,PSON=104400                              @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=251808,PSON=108000                              @000.000
RAMFIL TYPE=4SA,RECNO=03600,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=252808,PSON=111600                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=253808,PSON=122000                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=253908,PSON=122360                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254008,PSON=122720                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254108,PSON=123080                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254208,PSON=123440                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254308,PSON=123800                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254408,PSON=124160                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254508,PSON=124520                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254608,PSON=124880                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254708,PSON=125240                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254808,PSON=125600                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=254908,PSON=125960                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=255008,PSON=126320                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=255108,PSON=126680                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=255208,PSON=127040                              @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X

```

BASE=255308,PSON=127400	@000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255408,PSON=127760	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255508,PSON=128120	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255608,PSON=128480	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255708,PSON=128840	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255808,PSON=129200	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=255908,PSON=129560	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256008,PSON=129920	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256108,PSON=130280	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256208,PSON=130640	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256308,PSON=131000	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256408,PSON=131360	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256508,PSON=131720	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256608,PSON=132080	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256708,PSON=132440	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256808,PSON=132800	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=256908,PSON=133160	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257008,PSON=133520	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257108,PSON=133880	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257208,PSON=134240	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257308,PSON=134600	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257408,PSON=134960	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257508,PSON=135320	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257608,PSON=135680	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257708,PSON=136040	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257808,PSON=136400	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=257908,PSON=136760	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258008,PSON=137120	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258108,PSON=137480	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258208,PSON=137840	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258308,PSON=138200	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258408,PSON=138560	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258508,PSON=138920	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258608,PSON=139280	X @000.000

RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258708,PSON=139640	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258808,PSON=140000	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=258908,PSON=140360	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259008,PSON=140720	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259108,PSON=141080	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259208,PSON=141440	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259308,PSON=141800	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259408,PSON=142160	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259508,PSON=142520	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259608,PSON=142880	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259708,PSON=143240	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259808,PSON=143600	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=259908,PSON=143960	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260008,PSON=144320	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260108,PSON=144680	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260208,PSON=145040	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260308,PSON=145400	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260408,PSON=145760	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260508,PSON=146120	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260608,PSON=146480	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260708,PSON=146840	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260808,PSON=147200	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=260908,PSON=147560	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=261008,PSON=147920	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=261108,PSON=148280	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=261208,PSON=148640	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=261308,PSON=149000	X @000.000
RAMFIL TYPE=4SA,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=261408,PSON=149360	X @000.000
*	
*****	
*	
* NEXT BASE ADDRESS IS 261508	
*	
* 4K DUPED FIXED FILE RECORDS	
* 80424 RECORDS TOTAL	
*	
RAMFIL TYPE=4SA,RECNO=017208,RECID=SPARE,DUPE=YES,BASE=261508	@000.000
RAMFIL TYPE=4SA,RECNO=00156,RECID=#BRIDCOR,DUPE=YES,PRIOR=2	@000.000
RAMFIL TYPE=4SA,RECNO=30000,RECID=#XPRG1,DUPE=YES,PRIOR=9	@000.000

```

RAMFIL TYPE=4SA,RECNO=30000,RECID=#XPRG2,DUPE=YES,PRIOR=9      @000.000
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,D,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00420,RECID=#RV1RU,DUPE=YES,PRIOR=2,USER=(*,0,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,B,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,C,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,D,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,E,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,Z,*)
RAMFIL TYPE=4SA,RECNO=00090,RECID=#RV2RU,DUPE=YES,PRIOR=2,USER=(*,0,*)

```

\*

\*\*\*\*\*

\*

\* START DEVICE B

\*

\* NEXT BASE ADDRESS IS 000002

\*

\* 4K DUPED FIXED FILE RECORDS

\* 283176 TOTAL RECORDS

\*

\* VTOC - 333914

\*

```

RAMFIL TYPE=4SB,RECNO=06000,RECID=#PROG1,DUPE=YES,BAND=2285,      X
      UFTI4=((61,134),(63,06)),      X
      UFTI5=((51,134),(53,06)),BASE=00002      @000.000
RAMFIL TYPE=4SB,RECNO=06000,RECID=#PROG2,DUPE=YES,BAND=4032,      X
      UFTI4=((61,135),(63,07)),      X
      UFTI5=((51,135),(53,07))      @000.000
RAMFIL TYPE=4SB,RECNO=06000,RECID=#PROG3,DUPE=YES,BAND=4033,      X
      UFTI4=(0,11),UFTI5=(0,12),EQU=215      @000.000
RAMFIL TYPE=4SB,RECNO=6000,RECID=#PROG4,DUPE=YES,BAND=4034,      X
      UFTI4=(0,21),UFTI5=(0,22),EQU=221      @000.000
RAMFIL TYPE=4SB,RECNO=6000,RECID=#PROG5,DUPE=YES,BAND=4035,      X
      UFTI4=((62,58),(62,62),(63,32)),      X
      UFTI5=((52,58),(52,62)),EQU=418      @000.000
RAMFIL TYPE=4SB,RECNO=6000,RECID=#PROG6,DUPE=YES,BAND=4044,      X
      UFTI4=((62,59),(62,63),(63,33)),      X
      UFTI5=((52,59),(52,63)),EQU=419      @000.000
RAMFIL TYPE=4SB,RECNO=6000,RECID=#PROG7,DUPE=YES,BAND=4045,      X
      UFTI4=((62,60),(62,64),(63,34)),      X
      UFTI5=((52,60),(52,64)),EQU=424      @000.000
RAMFIL TYPE=4SB,RECNO=6000,RECID=#PROG8,DUPE=YES,BAND=4046,      X
      UFTI4=((62,61),(62,65),(63,35)),      X
      UFTI5=((52,61),(52,65)),EQU=431      @000.000
RAMFIL TYPE=4SB,RECNO=00234,RECID=#KSA5,DUPE=YES,BAND=4047,      X
      UFTI4=(61,168),UFTI5=(51,168),EQU=432      @000.000
RAMFIL TYPE=4SB,RECNO=00066,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00234,RECID=#KSA6,DUPE=YES,BAND=4048,      X
      UFTI4=(61,167),UFTI5=(51,167),EQU=425      @000.000
RAMFIL TYPE=4SB,RECNO=00066,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00234,RECID=#KSA7,DUPE=YES,BAND=4049,      X
      UFTI4=(61,104),UFTI5=(51,104),EQU=426      @000.000
RAMFIL TYPE=4SB,RECNO=00066,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00234,RECID=#KSA8,DUPE=YES,BAND=4050,      X
      UFTI4=(61,169),UFTI5=(51,169),EQU=427      @000.000
RAMFIL TYPE=4SB,RECNO=00066,RECID=SPARE,DUPE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00050,RECID=#PVR5,DUPE=YES,BAND=4051,      X
      UFTI4=(61,170),UFTI5=(51,170),EQU=428      @000.000
RAMFIL TYPE=4SB,RECNO=00050,RECID=#PVR6,DUPE=YES,BAND=4052,      X
      UFTI4=(61,165),UFTI5=(51,165),EQU=429      @000.000
RAMFIL TYPE=4SB,RECNO=00050,RECID=#PVR7,DUPE=YES,BAND=4021,      X
      UFTI4=(62,31),UFTI5=(52,31),EQU=400      @000.000
RAMFIL TYPE=4SB,RECNO=00050,RECID=#PVR8,DUPE=YES,BAND=4022,      X
      UFTI4=(62,32),UFTI5=(52,32),EQU=401      @000.000

```



```

RAMFIL TYPE=4SB,RECNO=02000,RECID=#OLD5,DUPE=YES,BAND=4023,EQU=402,    X
      UFTI4=((62,33),(61,427),(61,431),(63,250)),                        X
      UFTI5=(52,33)                                                        @000.000
RAMFIL TYPE=4SB,RECNO=02000,RECID=#OLD6,DUPE=YES,BAND=4024,EQU=403,    X
      UFTI4=((62,34),(61,428),(61,432),(63,251)),                        X
      UFTI5=(52,34)                                                        @000.000
RAMFIL TYPE=4SB,RECNO=02000,RECID=#OLD7,DUPE=YES,BAND=4025,EQU=404,    X
      UFTI4=((62,35),(61,429),(61,433),(63,252)),                        X
      UFTI5=(52,35)                                                        @000.000
RAMFIL TYPE=4SB,RECNO=02000,RECID=#OLD8,DUPE=YES,BAND=4026,EQU=405,    X
      UFTI4=((62,36),(61,430),(61,434),(63,253)),                        X
      UFTI5=(52,36)                                                        @000.000
RAMFIL TYPE=4SB,RECNO=24000,RECID=#XPRG5,DUPE=YES,BAND=4027,            X
      UFTI4=((62,54),(62,66),(63,36),(63,42)),                          X
      UFTI5=((52,54),(52,66),(53,12),(53,18)),EQU=406 @000.000
RAMFIL TYPE=4SB,RECNO=24000,RECID=#XPRG6,DUPE=YES,BAND=4029,            X
      UFTI4=((62,55),(62,67),(63,37),(63,43)),                          X
      UFTI5=((52,55),(52,67),(53,13),(53,19)),EQU=407 @000.000
RAMFIL TYPE=4SB,RECNO=24000,RECID=#XPRG7,DUPE=YES,BAND=4030,            X
      UFTI4=((62,56),(62,68),(63,38),(63,44)),                          X
      UFTI5=((52,56),(52,68),(53,14),(53,20)),EQU=408 @000.000
RAMFIL TYPE=4SB,RECNO=24000,RECID=#XPRG8,DUPE=YES,BAND=4031,            X
      UFTI4=((62,57),(62,69),(63,39),(63,45)),                          X
      UFTI5=((52,57),(52,69),(53,15),(53,21)),EQU=409 @000.000
RAMFIL TYPE=4SB,RECNO=17683,RECID=#XPRG1,DUPE=YES,PRIOR=8 @000.000
RAMFIL TYPE=4SB,RECNO=17683,RECID=#XPRG2,DUPE=YES,PRIOR=8 @000.000
RAMFIL TYPE=4SB,RECNO=17683,RECID=#XPRG3,DUPE=YES,PRIOR=8 @000.000
RAMFIL TYPE=4SB,RECNO=17683,RECID=#XPRG4,DUPE=YES,PRIOR=8 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG1,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG2,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG3,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG4,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG5,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG6,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG7,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#PROG8,DUPE=YES,PRIOR=2 @000.000
RAMFIL TYPE=4SB,RECNO=000020,RECID=SPARE,DUPE=YES @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG1,DUPE=YES,PRIOR=3,BASE=64500 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG2,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG3,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG4,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG5,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG6,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG7,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#PROG8,DUPE=YES,PRIOR=3 @000.000
RAMFIL TYPE=4SB,RECNO=03760,RECID=SPARE,DUPE=YES
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD1,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD2,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD3,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD4,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD5,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD6,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD7,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=01000,RECID=#OLD8,DUPE=YES,PRIOR=2
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,982),UFTI5=(51,982),EQU=308,USER=(HPN,F,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,983),UFTI5=(51,983),EQU=308,USER=(HPN,G,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,984),UFTI5=(51,984),EQU=308,USER=(HPN2,F,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,985),UFTI5=(51,985),EQU=308,USER=(HPN2,G,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,498),UFTI5=(51,498),EQU=308,USER=(HPN,H,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X
      UFTI4=(61,499),UFTI5=(51,499),EQU=308,USER=(HPN,I,*)              X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,                    X

```

```

      UFTI4=(61,500),UFTI5=(51,500),EQU=308,USER=(HPN,J,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES, X
      UFTI4=(61,501),UFTI5=(51,501),EQU=308,USER=(HPN,K,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES, X
      UFTI4=(61,502),UFTI5=(51,502),EQU=308,USER=(HPN2,H,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES, X
      UFTI4=(61,503),UFTI5=(51,503),EQU=308,USER=(HPN2,I,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES, X
      UFTI4=(61,504),UFTI5=(51,504),EQU=308,USER=(HPN2,J,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES, X
      UFTI4=(61,505),UFTI5=(51,505),EQU=308,USER=(HPN2,K,*)
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,613),UFTI5=(51,613),USER=(HPN,L,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,614),UFTI5=(51,614),USER=(HPN2,L,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,615),UFTI5=(51,615),USER=(HPN,M,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,616),UFTI5=(51,616),USER=(HPN2,M,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,617),UFTI5=(51,617),USER=(HPN,N,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,618),UFTI5=(51,618),USER=(HPN2,N,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,619),UFTI5=(51,619),USER=(HPN,O,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,620),UFTI5=(51,620),USER=(HPN2,O,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,621),UFTI5=(51,621),USER=(HPN,P,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,622),UFTI5=(51,622),USER=(HPN2,P,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,623),UFTI5=(51,623),USER=(HPN,Q,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,624),UFTI5=(51,624),USER=(HPN2,Q,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,625),UFTI5=(51,625),USER=(HPN,R,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,626),UFTI5=(51,626),USER=(HPN2,R,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,627),UFTI5=(51,627),USER=(HPN,S,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,628),UFTI5=(51,628),USER=(HPN2,S,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,629),UFTI5=(51,629),USER=(HPN,T,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,630),UFTI5=(51,630),USER=(HPN2,T,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,631),UFTI5=(51,631),USER=(HPN,U,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,632),UFTI5=(51,632),USER=(HPN2,U,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,633),UFTI5=(51,633),USER=(HPN,V,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,634),UFTI5=(51,634),USER=(HPN2,V,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,635),UFTI5=(51,635),USER=(HPN,W,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,636),UFTI5=(51,636),USER=(HPN2,W,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,838),UFTI5=(51,838),USER=(HPN,X,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,839),UFTI5=(51,839),USER=(HPN2,X,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,840),UFTI5=(51,840),USER=(HPN,Y,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, X
      UFTI4=(61,841),UFTI5=(51,841),USER=(HPN2,Y,*) @000.000

```

RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,842),UFTI5=(51,842),USER=(HPN,1,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,843),UFTI5=(51,843),USER=(HPN2,1,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,844),UFTI5=(51,844),USER=(HPN,2,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,845),UFTI5=(51,845),USER=(HPN2,2,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,846),UFTI5=(51,846),USER=(HPN,3,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,847),UFTI5=(51,847),USER=(HPN2,3,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,848),UFTI5=(51,848),USER=(HPN,4,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,849),UFTI5=(51,849),USER=(HPN2,4,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,850),UFTI5=(51,850),USER=(HPN,5,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,851),UFTI5=(51,851),USER=(HPN2,5,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,852),UFTI5=(51,852),USER=(HPN,6,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDSA8,DUPE=YES,EQU=308, UFTI4=(61,853),UFTI5=(51,853),USER=(HPN2,6,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,986),UFTI5=(51,986),EQU=309,USER=(HPN,F,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,987),UFTI5=(51,987),EQU=309,USER=(HPN,G,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,988),UFTI5=(51,988),EQU=309,USER=(HPN2,F,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,989),UFTI5=(51,989),EQU=309,USER=(HPN2,G,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,506),UFTI5=(51,506),EQU=309,USER=(HPN,H,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,507),UFTI5=(51,507),EQU=309,USER=(HPN,I,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,508),UFTI5=(51,508),EQU=309,USER=(HPN,J,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,509),UFTI5=(51,509),EQU=309,USER=(HPN,K,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,510),UFTI5=(51,510),EQU=309,USER=(HPN2,H,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,511),UFTI5=(51,511),EQU=309,USER=(HPN2,I,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,512),UFTI5=(51,512),EQU=309,USER=(HPN2,J,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES, UFTI4=(61,513),UFTI5=(51,513),EQU=309,USER=(HPN2,K,*)	X
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,637),UFTI5=(51,637),USER=(HPN,L,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,638),UFTI5=(51,638),USER=(HPN2,L,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,639),UFTI5=(51,639),USER=(HPN,M,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,640),UFTI5=(51,640),USER=(HPN2,M,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,641),UFTI5=(51,641),USER=(HPN,N,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,642),UFTI5=(51,642),USER=(HPN2,N,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,643),UFTI5=(51,643),USER=(HPN,O,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,644),UFTI5=(51,644),USER=(HPN2,O,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, UFTI4=(61,645),UFTI5=(51,645),USER=(HPN,P,*)	X @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309,	X



```

      UFTI4=(61,646),UFTI5=(51,646),USER=(HPN2,P,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,647),UFTI5=(51,647),USER=(HPN,Q,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,648),UFTI5=(51,648),USER=(HPN2,Q,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,649),UFTI5=(51,649),USER=(HPN,R,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,650),UFTI5=(51,650),USER=(HPN2,R,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,651),UFTI5=(51,651),USER=(HPN,S,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,652),UFTI5=(51,652),USER=(HPN2,S,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,653),UFTI5=(51,653),USER=(HPN,T,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,654),UFTI5=(51,654),USER=(HPN2,T,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,655),UFTI5=(51,655),USER=(HPN,U,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,656),UFTI5=(51,656),USER=(HPN2,U,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,657),UFTI5=(51,657),USER=(HPN,V,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,658),UFTI5=(51,658),USER=(HPN2,V,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,659),UFTI5=(51,659),USER=(HPN,W,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,660),UFTI5=(51,660),USER=(HPN2,W,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,854),UFTI5=(51,854),USER=(HPN,X,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,855),UFTI5=(51,855),USER=(HPN2,X,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,856),UFTI5=(51,856),USER=(HPN,Y,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,857),UFTI5=(51,857),USER=(HPN2,Y,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,858),UFTI5=(51,858),USER=(HPN,1,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,859),UFTI5=(51,859),USER=(HPN2,1,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,860),UFTI5=(51,860),USER=(HPN,2,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,861),UFTI5=(51,861),USER=(HPN2,2,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,862),UFTI5=(51,862),USER=(HPN,3,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,863),UFTI5=(51,863),USER=(HPN2,3,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,864),UFTI5=(51,864),USER=(HPN,4,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,865),UFTI5=(51,865),USER=(HPN2,4,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,866),UFTI5=(51,866),USER=(HPN,5,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,867),UFTI5=(51,867),USER=(HPN2,5,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,868),UFTI5=(51,868),USER=(HPN,6,*) @000.000
RAMFIL TYPE=4SB,RECNO=00129,RECID=#BRIDTB8,DUPE=YES,EQU=309, X
      UFTI4=(61,869),UFTI5=(51,869),USER=(HPN2,6,*) @000.000
RAMFIL TYPE=4SB,RECNO=01824,RECID=SPARE,DUPE=YES @000.000

```

\*

\*\*\*\*\*

\*

\* NEXT BASE ADDRESS IS 078611

\*

```

*      4K LONG TERM DUPED POOL RECORDS (4DP-B)
*      12 DIRECTORIES OF 24 ADDRESSES EACH
*      1 DIRECTORY OF 304 ADDRESSES EACH
*      1 DIRECTORY OF 8000 ADDRESSES EACH
*      8592 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200000, OVERRIDE=YES,      X
      BASE=78610      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200024, OVERRIDE=YES,      X
      BASE=78611      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200048, OVERRIDE=YES,      X
      BASE=78612      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200072, OVERRIDE=YES,      X
      BASE=78613      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200096, OVERRIDE=YES,      X
      BASE=78614      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200120, OVERRIDE=YES,      X
      BASE=78700      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200144, OVERRIDE=YES,      X
      BASE=78701      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200168, OVERRIDE=YES,      X
      BASE=78702      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200192, OVERRIDE=YES,      X
      BASE=78703      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200216, OVERRIDE=YES,      X
      BASE=78704      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200240, OVERRIDE=YES,      X
      BASE=78705      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200264, OVERRIDE=YES,      X
      BASE=78706      @000.000
RAMFIL TYPE=4SB,RECNO=8304,RECID=POOL,DUPE=YES,POLID=LT,      X
      PSON=200288, OVERRIDE=YES,      X
      BASE=78707      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 081009
*
*      4K SHORT TERM POOL RECORDS (4ST-B)
*      12 DIRECTORIES OF 24 ADDRESSES EACH
*      1 DIRECTORY OF 304 ADDRESSES EACH
*      1 DIRECTORY OF 8000 ADDRESSES EACH
*      8592 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST,      X
      PSON=201000, OVERRIDE=YES,      X
      BASE=81008      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST,      X
      PSON=201024, OVERRIDE=YES,      X
      BASE=81009      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST,      X
      PSON=201048, OVERRIDE=YES,      X
      BASE=81010      @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST,      X
      PSON=201072, OVERRIDE=YES,      X

```

```

BASE=81011 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201096,OVERRIDE=YES, X
      BASE=81012 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201120,OVERRIDE=YES, X
      BASE=81013 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201144,OVERRIDE=YES, X
      BASE=81014 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201168,OVERRIDE=YES, X
      BASE=81100 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201192,OVERRIDE=YES, X
      BASE=81101 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201216,OVERRIDE=YES, X
      BASE=81102 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201240,OVERRIDE=YES, X
      BASE=81103 @000.000
RAMFIL TYPE=4SB,RECNO=24,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201264,OVERRIDE=YES, X
      BASE=81104 @000.000
RAMFIL TYPE=4SB,RECNO=8304,RECID=POOL,DUPE=NO,POLID=ST, X
      PSON=201288,OVERRIDE=YES, X
      BASE=81105 @000.000
*
*****
*
* NEXT BASE ADDRESS IS 083406
*
* 4K DUPED FIXED FILE RECORDS
* 73608 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=004800,RECID=SPARE,DUPE=YES,BASE=83406 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG7,DUPE=YES,BASE=084711, X
      UFTI4=(63,256),UFTI5=(53,256),EQU=284 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG8,DUPE=YES,BASE=086111, X
      UFTI4=(63,257),UFTI5=(53,257),EQU=285 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG9,DUPE=YES,BASE=087511, X
      UFTI4=(63,435),UFTI5=(53,435),EQU=286 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG10,DUPE=YES,BASE=088911, X
      UFTI4=(63,436),UFTI5=(53,436),EQU=287 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG11,DUPE=YES,BASE=090311, X
      UFTI4=(63,437),UFTI5=(53,437),EQU=288 @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG1,DUPE=YES,BASE=91711, X
      BAND=3709,UFTI4=((63,24),(63,50),(63,51),(63,52)), X
      UFTI5=((51,214),(52,50),(52,51),(52,52),(52,53),(52,70))X
      ,EQU=359 @000.000
RAMFIL TYPE=4SB,RECNO=2520,RECID=#RLOG2,DUPE=YES,BASE=93111, X
      UFTI4=(63,25),UFTI5=((51,215),(51,310),(51,315)), X
      EQU=360,BAND=3710 @000.000
RAMFIL TYPE=4SB,RECNO=2520,RECID=#RLOG3,DUPE=YES,BASE=94511, X
      UFTI4=(63,26),UFTI5=((51,216),(51,311),(51,316)), X
      EQU=361,BAND=3711 @000.000
RAMFIL TYPE=4SB,RECNO=2520,RECID=#RLOG4,DUPE=YES,BASE=95911, X
      UFTI4=(63,27),UFTI5=((51,217),(51,312),(51,317)), X
      EQU=362,BAND=3712 @000.000
RAMFIL TYPE=4SB,RECNO=2520,RECID=#RLOG5,DUPE=YES,BASE=97311, X
      UFTI4=(63,28),UFTI5=((51,218),(51,313),(51,318)), X
      EQU=363,BAND=3713 @000.000
RAMFIL TYPE=4SB,RECNO=2520,RECID=#RLOG6,DUPE=YES,BAND=3694,EQU=364, X
      BASE=98711, X
      UFTI4=(63,29),UFTI5=((51,219),(51,314),(51,319))
RAMFIL TYPE=4SB,RECNO=00180,RECID=SPARE,DUPE=YES,BASE=100111 @000.000

```

```

RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR1,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR2,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR3,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR4,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR5,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR6,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR7,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=00030,RECID=#PVR8,DUPE=YES,PRIOR=2      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#MAIL01,DUPE=YES,BAND=1008,EQU=280, X
      UFTI4=((61,435),(61,439)),UFTI5=(51,435)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#MAIL02,DUPE=YES,BAND=1009,EQU=281, X
      UFTI4=((61,436),(61,440)),UFTI5=(51,436)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#MAIL03,DUPE=YES,BAND=1010,EQU=282, X
      UFTI4=((61,437),(61,441)),UFTI5=(51,437)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#MAIL04,DUPE=YES,BAND=1011,EQU=283, X
      UFTI4=((61,438),(61,442)),UFTI5=(51,438)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG1,DUPE=YES,BAND=1000,EQU=260, X
      UFTI4=((61,410),(61,418)),UFTI5=(51,410)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG2,DUPE=YES,BAND=1001,EQU=261, X
      UFTI4=((61,411),(61,419)),UFTI5=(51,411)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG3,DUPE=YES,BAND=1002,EQU=262, X
      UFTI4=((61,412),(61,421)),UFTI5=(51,412)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG4,DUPE=YES,BAND=1003,EQU=263, X
      UFTI4=((61,413),(61,422)),UFTI5=(51,413)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG5,DUPE=YES,BAND=1004,EQU=264, X
      UFTI4=((61,414),(61,423)),UFTI5=(51,414)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG6,DUPE=YES,BAND=1005,EQU=265, X
      UFTI4=((61,415),(61,424)),UFTI5=(51,415)      @000.000
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG7,DUPE=YES,BAND=1006,EQU=266, X
      UFTI4=((61,416),(61,425)),UFTI5=(51,416)
RAMFIL TYPE=4SB,RECNO=01000,RECID=#APRG8,DUPE=YES,BAND=1007,EQU=267, X
      UFTI4=((61,417),(61,426)),UFTI5=(51,417)
RAMFIL TYPE=4SB,RECNO=000948,RECID=SPARE,DUPE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 103813
*
* 4K SHORT TERM POOL RECORDS (4ST-B)
* 50 DIRECTORIES OF 480 ADDRESSES EACH
* 1 DIRECTORY OF 632 ADDRESSES EACH
* 5 DIRECTORIES OF 8000 ADDRESSES EACH
* 32160 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=103813,PSON=209592,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104003,PSON=210072,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104108,PSON=210552,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104213,PSON=211032,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104403,PSON=211512,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104508,PSON=211992,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104613,PSON=212472,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104803,PSON=212952,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=104908,PSON=213432,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=105013,PSON=213912,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=40632,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=105203,PSON=214392,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X

```

BASE=116501,PSON=255024,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=116606,PSON=255504,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=116711,PSON=255984,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=116901,PSON=256464,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117006,PSON=256944,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117111,PSON=257424,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117301,PSON=257904,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117406,PSON=258384,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117511,PSON=258864,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117701,PSON=259344,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117806,PSON=259824,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=117911,PSON=260304,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118101,PSON=260784,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118206,PSON=261264,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118311,PSON=261744,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118501,PSON=262224,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118606,PSON=262704,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118711,PSON=263184,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=118901,PSON=263664,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119006,PSON=264144,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119111,PSON=264624,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119301,PSON=265104,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119406,PSON=265584,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119511,PSON=266064,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119701,PSON=266544,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119806,PSON=267024,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=119911,PSON=267504,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120101,PSON=267984,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120206,PSON=268464,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120311,PSON=268944,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120501,PSON=269424,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120606,PSON=269904,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120711,PSON=270384,OVERRIDE=YES	@000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,	X
BASE=120901,PSON=270864,OVERRIDE=YES	@000.000

```

RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121006,PSON=271344,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121111,PSON=271824,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121301,PSON=272304,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121406,PSON=272784,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121511,PSON=273264,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00480,RECID=POOL,DUPE=NO,POLID=ST,      X
      BASE=121701,PSON=273744,OVERRIDE=YES      @000.000
*
*****
*
* NEXT BASE ADDRESS IS 103813
*
* 4K LONG TERM DUPED POOL RECORDS (4DP-B)
* 71 DIRECTORIES OF 1440 ADDRESSES EACH
* 1 DIRECTORY OF 1448 ADDRESSES EACH
* 10 DIRECTORIES OF 8000 ADDRESSES EACH
* 183688 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=25440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=121806,PSON=316928      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=128901,PSON=342368      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=129301,PSON=343808      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=129701,PSON=345248      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=130101,PSON=346688      @000.000
RAMFIL TYPE=4SB,RECNO=25440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=130501,PSON=348128      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=137511,PSON=373568      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=137911,PSON=375008      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=138311,PSON=376448      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=138711,PSON=377888      @000.000
RAMFIL TYPE=4SB,RECNO=25440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=139111,PSON=379328      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=146206,PSON=404768      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=146606,PSON=406208      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=147006,PSON=407648      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=147406,PSON=409088      @000.000
RAMFIL TYPE=4SB,RECNO=08448,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=147806,PSON=410528      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=150113,PSON=418976      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=150513,PSON=420416      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=150913,PSON=421856      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=151313,PSON=423296      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=151713,PSON=424736      @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,      X
      BASE=152113,PSON=426176      @000.000

```



RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=152513,PSON=427616	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=152913,PSON=429056	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=153313,PSON=430496	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=153713,PSON=431936	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=154113,PSON=433376	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=154513,PSON=434816	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=154913,PSON=436256	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=155313,PSON=437696	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=155713,PSON=439136	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=156113,PSON=440576	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=156513,PSON=442016	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=156913,PSON=443456	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=157313,PSON=444896	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=157713,PSON=446336	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=158113,PSON=447776	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=158513,PSON=449216	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=158913,PSON=450656	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=159313,PSON=452096	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=159713,PSON=453536	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=160113,PSON=454976	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=160513,PSON=456416	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=160913,PSON=457856	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=161313,PSON=459296	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=161713,PSON=460736	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=162113,PSON=462176	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=162513,PSON=463616	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=162913,PSON=465056	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=163313,PSON=466496	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=163713,PSON=467936	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=164113,PSON=469376	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=164513,PSON=470816	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=164913,PSON=472256	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT, BASE=165313,PSON=473696	X 0000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,	X

```

      BASE=165713,PSON=475136                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=166113,PSON=476576                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=166513,PSON=478016                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=166913,PSON=479456                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=167313,PSON=480896                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=167713,PSON=482336                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=168113,PSON=483776                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=168513,PSON=485216                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=168913,PSON=486656                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=169313,PSON=488096                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=169713,PSON=489536                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=170113,PSON=490976                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=170513,PSON=492416                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=170913,PSON=493856                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=171313,PSON=495296                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=171713,PSON=496736                                @000.000
RAMFIL TYPE=4SB,RECNO=01440,RECID=POOL,DUPE=YES,POLID=LT,    X
      BASE=172113,PSON=498176                                @000.000
*
*****
*
* NEXT BASE ADDRESS IS 172513
*
*      4K DUPED FIXED RECORDS (VERTICALLY ALLOCATED)
*      105840 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG12,DUPE=YES,BASE=172513,    X
      UFTI4=(63,438),UFTI5=(53,438),EQU=289                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG13,DUPE=YES,BASE=173913,    X
      UFTI4=(63,697),UFTI5=(53,697),EQU=318                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG14,DUPE=YES,BASE=175313,    X
      UFTI4=(63,698),UFTI5=(53,698),EQU=319                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG15,DUPE=YES,BASE=176713,    X
      UFTI4=(63,699),UFTI5=(53,699),EQU=320                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG16,DUPE=YES,BASE=178113,    X
      UFTI4=(63,700),UFTI5=(53,700),EQU=321                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG17,DUPE=YES,BASE=179513,    X
      UFTI4=(63,701),UFTI5=(53,701),EQU=322                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG18,DUPE=YES,BASE=180913,    X
      UFTI4=(63,702),UFTI5=(53,702),EQU=323                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG19,DUPE=YES,BASE=182313,    X
      UFTI4=(63,703),UFTI5=(53,703),EQU=324                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG20,DUPE=YES,BASE=183713,    X
      UFTI4=(63,704),UFTI5=(53,704),EQU=325                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG21,DUPE=YES,BASE=185113,    X
      UFTI4=(63,705),UFTI5=(53,705),EQU=326                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG22,DUPE=YES,BASE=186513,    X
      UFTI4=(63,706),UFTI5=(53,706),EQU=327                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG23,DUPE=YES,BASE=187913,    X
      UFTI4=(63,707),UFTI5=(53,707),EQU=328                @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG24,DUPE=YES,BASE=189313,    X
      UFTI4=(63,708),UFTI5=(53,708),EQU=329                @000.000

```



```

RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG25,DUPE=YES,BASE=190713,      X
      UFTI4=(63,897),UFTI5=(53,897),EQU=330      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG26,DUPE=YES,BASE=192113,      X
      UFTI4=(63,898),UFTI5=(53,898),EQU=348      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG27,DUPE=YES,BASE=193513,      X
      UFTI4=(63,899),UFTI5=(53,899),EQU=349      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG28,DUPE=YES,BASE=194913,      X
      UFTI4=(63,900),UFTI5=(53,900),EQU=350      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG29,DUPE=YES,BASE=196313,      X
      UFTI4=(63,901),UFTI5=(53,901),EQU=351      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG30,DUPE=YES,BASE=197713,      X
      UFTI4=(63,902),UFTI5=(53,902),EQU=352      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG31,DUPE=YES,BASE=199113,      X
      UFTI4=(63,903),UFTI5=(53,903),EQU=353      @000.000
RAMFIL TYPE=4SB,RECNO=02520,RECID=#RLOG32,DUPE=YES,BASE=200513,      X
      UFTI4=(63,904),UFTI5=(53,904),EQU=354      @000.000

```

\*

\*\*\*\*\*

\*

\* NEXT BASE ADDRESS IS 201913

\*

\* 4K FARF6 LONG TERM DUPED POOL RECORDS (4D6-B)

\* 96 DIRECTORIES OF 360 ADDRESSES EACH

\* 4 DIRECTORIES OF 720 ADDRESSES EACH

\* 37440 RECORDS TOTAL

\*

```

RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=201913,PSON=115200,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202013,PSON=115560,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202113,PSON=115920,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202213,PSON=116280,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00720,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202313,PSON=116640,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00720,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202513,PSON=117360,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202713,PSON=118080,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202813,PSON=118440,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=202913,PSON=118800,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203013,PSON=119160,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00720,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203113,PSON=119520,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00720,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203313,PSON=120240,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203513,PSON=120960,OVERRIDE=YES      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203613,PSON=149720      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203713,PSON=150080      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203813,PSON=150440      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=203913,PSON=150800      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=204013,PSON=151160      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=204113,PSON=151520      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X
      BASE=204213,PSON=151880      @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,          X

```

BASE=204313,PSON=152240	@000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204413,PSON=152600	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204513,PSON=152960	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204613,PSON=153320	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204713,PSON=153680	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204813,PSON=154040	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=204913,PSON=154400	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205013,PSON=154760	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205113,PSON=155120	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205213,PSON=155480	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205313,PSON=155840	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205413,PSON=156200	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205513,PSON=156560	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205613,PSON=156920	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205713,PSON=157280	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205813,PSON=157640	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=205913,PSON=158000	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206013,PSON=158360	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206113,PSON=158720	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206213,PSON=159080	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206313,PSON=159440	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206413,PSON=159800	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206513,PSON=160160	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206613,PSON=160520	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206713,PSON=160880	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206813,PSON=161240	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=206913,PSON=161600	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207013,PSON=161960	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207113,PSON=162320	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207213,PSON=162680	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207313,PSON=163040	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207413,PSON=163400	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207513,PSON=163760	X @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207613,PSON=164120	X @000.000

RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207713,PSON=164480	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207813,PSON=164840	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=207913,PSON=165200	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208013,PSON=165560	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208113,PSON=165920	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208213,PSON=166280	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208313,PSON=166640	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208413,PSON=167000	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208513,PSON=167360	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208613,PSON=167720	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208713,PSON=168080	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208813,PSON=168440	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=208913,PSON=168800	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209013,PSON=169160	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209113,PSON=169520	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209213,PSON=169880	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209313,PSON=170240	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209413,PSON=170600	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209513,PSON=170960	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209613,PSON=171320	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209713,PSON=171680	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209813,PSON=172040	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=209913,PSON=172400	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210013,PSON=172760	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210113,PSON=173120	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210213,PSON=173480	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210313,PSON=173840	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210413,PSON=174200	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210513,PSON=174560	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210613,PSON=174920	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210713,PSON=175280	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210813,PSON=175640	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT, BASE=210913,PSON=176000	X 0000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,	X

```

      BASE=211013,PSON=176360                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211113,PSON=176720                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211213,PSON=177080                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211313,PSON=177440                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211413,PSON=177800                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211513,PSON=178160                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211613,PSON=178520                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211713,PSON=178880                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211813,PSON=179240                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=211913,PSON=179600                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=212013,PSON=179960                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=212113,PSON=180320                                @000.000
RAMFIL TYPE=4SB,RECNO=00360,RECID=POOL6,DUPE=YES,POLID=LT,    X
      BASE=212213,PSON=180680                                @000.000
*
*****
*
* NEXT BASE ADDRESS IS 212313
*
*      4K DUPED FIXED FILE RECORDS
*      436424 RECORDS TOTAL
*
RAMFIL TYPE=4SB,RECNO=317600,RECID=SPARE,DUPE=YES,BASE=212313 @000.000
RAMFIL TYPE=4SB,RECNO=14471,RECID=#IRCFDF,DUPE=YES,PRIOR=2    @000.000
RAMFIL TYPE=4SB,RECNO=14471,RECID=#IRCGDF,DUPE=YES,PRIOR=2    @000.000
RAMFIL TYPE=4SB,RECNO=14471,RECID=#IRCFDF,DUPE=YES,PRIOR=2    @000.000
RAMFIL TYPE=4SB,RECNO=31702,RECID=#IRCMDF,DUPE=YES,PRIOR=2    @000.000
RAMFIL TYPE=4SB,RECNO=07949,RECID=#IRIDIF,DUPE=YES,PRIOR=2    @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,98),UFTI5=(52,98),USER=(*,R,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,99),UFTI5=(52,99),USER=(*,S,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,80),UFTI5=(52,80),USER=(*,T,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,81),UFTI5=(52,81),USER=(*,U,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,82),UFTI5=(52,82),USER=(*,V,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,83),UFTI5=(52,83),USER=(*,W,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,84),UFTI5=(52,84),USER=(*,X,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,85),UFTI5=(52,85),USER=(*,Y,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,86),UFTI5=(52,86),USER=(*,1,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,87),UFTI5=(52,87),USER=(*,2,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,88),UFTI5=(52,88),USER=(*,3,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,89),UFTI5=(52,89),USER=(*,4,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,90),UFTI5=(52,90),USER=(*,5,*)                @000.000
RAMFIL TYPE=4SB,RECNO=00840,RECID=#RV1RU,DUPE=YES,EQU=462,    X
      UFTI4=(62,91),UFTI5=(52,91),USER=(*,6,*)                @000.000

```

```

RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD1,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD2,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD3,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD4,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD5,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD6,DUPE=YES,PRIOR=3      @000.000
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD7,DUPE=YES,PRIOR=3      @000.000
* ARFIE IS THE ROVER
RAMFIL TYPE=4SB,RECNO=03000,RECID=#OLD8,DUPE=YES,PRIOR=3      @000.000
RAMEND                                                           @000.000
*
*   VTOC - 333814
*
*
*   EJECT 1                                                         @000.000
*
* The following UFTix and BAND allocations are available
* BAND=0852
* BAND=4002
* BAND=2812
*
*****
*                               LINES - LINE TYPES IN NETWORK      *
*****
*
*****
*
*   THE FOLLOWING LINES ARE ORDERED BY THEIR RELATIVE              *
*   LINE NUMBERS IN AN OPERATIONAL SYSTEM.                         *
*   D O N O T R E O R D E R   T H E S E   P A R A M E T E R S .   *
*   O R   K E N   W I L L   D O   I T   U P                       *
*
*****
*                               *****
*   LINES      MINLS=34,      STARTING SYMBOLIC LINE NUMBER      X
*               SLCAI=10,      NBR OF SYNCHRONOUS AI LINES        X
*               BSCLN=16,      NBR OF BISYNC LINES                X
*               BSCDRPS=155,    NBR OF BISYNC DROPS                X
*               BSCOP=(US,EB,MP,CT,PP), BISYNC OPTIONS            X
*               NLSLC=8,        NBR OF 3270 LOCAL LINES            X
*               PSLNS=2         NBR OF PSEUDO LINES
*
*   EJECT 1
*
*
*
*****
*                               NETWK - COMMUNICATION VARIABLES      *
*****
*
*   NETWK      D3270=YES,      ARE 3270 LINES DEFINED              X
*               L70MAX0=6680,  MAX 3270 LOCAL OUTPUT MESSAGE SIZE X
*               MAXDP=2,        MAX NBR OF 3705 DUMPS ON FILE      X
*               MAXEP=4,        MAX NBR OF EP LOAD MODULES         X
*               M3270=3600,     MAXIMUM 3270 MESSAGE LENGTH         X
*               NPATH=4,        NBR OF ALTERNATE PHYSICAL PATHS    X
*               N2703=3,        NUMBER OF ACTIVE 2703'S            X
*               SUBCH=(1C-4,1A-4), 3705 NATIVE SUBCHANNEL           X
*               DIAG=(Y8,Y8),     EXTENDED ADDRESSING DIAGNOSTICS  X
*               NUAT=2500        UNIQUE LN-IA-TA REQUIRED WITH AAA
*
*   EJECT 1
*
*
*
*****
*                               BBSAT - BSC STATION ADDRESS TABEL    *
*****
*
*   BBSAT      POLCH=C1,SELCH=E1

```

BBSAT POLCH=C2,SELCH=E2  
 BBSAT POLCH=41,SELCH=61,CHTYP=A  
 BBSAT POLCH=42,SELCH=62,CHTYP=A  
 BBSAT POLCH=C3,SELCH=E3  
 BBSAT POLCH=C4,SELCH=E4  
 BBSAT POLCH=C7,SELCH=E7  
 BBSAT POLCH=C6,SELCH=E6  
 BBSAT POLCH=D1,SELCH=F1  
 BBSAT POLCH=D2,SELCH=F2  
 BBSAT POLCH=D3,SELCH=F3  
 BBSAT POLCH=D4,SELCH=F4  
 BBSAT POLCH=D5,SELCH=F5  
 BBSAT POLCH=D6,SELCH=F6  
 BBSAT POLCH=D7,SELCH=F7  
 BBSAT POLCH=D8,SELCH=F8  
 BBSAT POLCH=D9,SELCH=F9

# EJECT 1

\*  
 \*  
 \*

\*\*\*\*\*

\* BSNT - BSC STATION NAME CONVERSION TABLE \*

\*\*\*\*\*

\*

\*OUT BSNT STANM=APES,CPUID=A,SYMLN=40,STANO=02,APPLN=CXJAFOR PAT Q  
 BSNT STANM=APES,CPUID=A,SYMLN=41,STANO=02,APPLN=CXJA  
 BSNT STANM=APES,CPUID=A,SYMLN=42,STANO=02,APPLN=CXJA  
 BSNT STANM=APES,CPUID=A,SYMLN=44,STANO=02,APPLN=CXJA  
 \*OUT BSNT STANM=ASSS,CPUID=B,SYMLN=40,STANO=02,APPLN=CXJBFOR PAT Q  
 BSNT STANM=ASSS,CPUID=B,SYMLN=41,STANO=02,APPLN=CXJB  
 BSNT STANM=ASSS,CPUID=B,SYMLN=42,STANO=02,APPLN=CXJB  
 BSNT STANM=ASSS,CPUID=B,SYMLN=44,STANO=02,APPLN=CXJB  
 \*OUT BSNT STANM=BATS,CPUID=A,SYMLN=40,STANO=03,APPLN=CXJAFOR PAT Q  
 BSNT STANM=BATS,CPUID=A,SYMLN=41,STANO=03,APPLN=CXJA  
 BSNT STANM=BATS,CPUID=A,SYMLN=42,STANO=03,APPLN=CXJA  
 BSNT STANM=BATS,CPUID=A,SYMLN=44,STANO=03,APPLN=CXJA  
 \*OUT BSNT STANM=BIRD,CPUID=B,SYMLN=40,STANO=03,APPLN=CXJBFOR PAT Q  
 BSNT STANM=BIRD,CPUID=B,SYMLN=41,STANO=03,APPLN=CXJB  
 BSNT STANM=BIRD,CPUID=B,SYMLN=42,STANO=03,APPLN=CXJB  
 BSNT STANM=BIRD,CPUID=B,SYMLN=44,STANO=03,APPLN=CXJB  
 \*OUT BSNT STANM=CATS,CPUID=A,SYMLN=40,STANO=04,APPLN=CXJAFOR PAT Q  
 BSNT STANM=CATS,CPUID=A,SYMLN=41,STANO=04,APPLN=CXJA  
 BSNT STANM=CATS,CPUID=A,SYMLN=42,STANO=04,APPLN=CXJA  
 BSNT STANM=CATS,CPUID=A,SYMLN=44,STANO=04,APPLN=CXJA  
 \*OUT BSNT STANM=COWS,CPUID=B,SYMLN=40,STANO=04,APPLN=CXJBFOR PAT Q  
 BSNT STANM=COWS,CPUID=B,SYMLN=41,STANO=04,APPLN=CXJB  
 BSNT STANM=COWS,CPUID=B,SYMLN=42,STANO=04,APPLN=CXJB  
 BSNT STANM=COWS,CPUID=B,SYMLN=44,STANO=04,APPLN=CXJB  
 \*OUT BSNT STANM=CRAB,CPUID=A,SYMLN=40,STANO=05,APPLN=CXJAFOR PAT Q  
 BSNT STANM=CRAB,CPUID=A,SYMLN=41,STANO=05,APPLN=CXJA  
 BSNT STANM=CRAB,CPUID=A,SYMLN=42,STANO=05,APPLN=CXJA  
 BSNT STANM=CRAB,CPUID=A,SYMLN=44,STANO=05,APPLN=CXJA  
 \*OUT BSNT STANM=CROW,CPUID=B,SYMLN=40,STANO=05,APPLN=CXJBFOR PAT Q  
 BSNT STANM=CROW,CPUID=B,SYMLN=41,STANO=05,APPLN=CXJB  
 BSNT STANM=CROW,CPUID=B,SYMLN=42,STANO=05,APPLN=CXJB  
 BSNT STANM=CROW,CPUID=B,SYMLN=44,STANO=05,APPLN=CXJB  
 \*OUT BSNT STANM=DEER,CPUID=A,SYMLN=40,STANO=06,APPLN=CXJAFOR PAT Q  
 BSNT STANM=DEER,CPUID=A,SYMLN=41,STANO=06,APPLN=CXJA  
 BSNT STANM=DEER,CPUID=A,SYMLN=42,STANO=06,APPLN=CXJA  
 BSNT STANM=DEER,CPUID=A,SYMLN=44,STANO=06,APPLN=CXJA  
 \*OUT BSNT STANM=DOGS,CPUID=B,SYMLN=40,STANO=06,APPLN=CXJBFOR PAT Q  
 BSNT STANM=DOGS,CPUID=B,SYMLN=41,STANO=06,APPLN=CXJB  
 BSNT STANM=DOGS,CPUID=B,SYMLN=42,STANO=06,APPLN=CXJB  
 BSNT STANM=DOGS,CPUID=B,SYMLN=44,STANO=06,APPLN=CXJB  
 \*OUT BSNT STANM=DUCK,CPUID=A,SYMLN=40,STANO=07,APPLN=CXJAFOR PAT Q  
 BSNT STANM=DUCK,CPUID=A,SYMLN=41,STANO=07,APPLN=CXJA  
 BSNT STANM=DUCK,CPUID=A,SYMLN=42,STANO=07,APPLN=CXJA



	BSNCT	STANM=DUCK,CPUID=A,SYMLN=44,STANO=07,APPLN=CXJA
*OUT	BSNCT	STANM=EELS,CPUID=B,SYMLN=40,STANO=07,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=EELS,CPUID=B,SYMLN=41,STANO=07,APPLN=CXJB
	BSNCT	STANM=EELS,CPUID=B,SYMLN=42,STANO=07,APPLN=CXJB
	BSNCT	STANM=EELS,CPUID=B,SYMLN=44,STANO=07,APPLN=CXJB
*OUT	BSNCT	STANM=EWES,CPUID=A,SYMLN=40,STANO=08,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=EWES,CPUID=A,SYMLN=41,STANO=08,APPLN=CXJA
	BSNCT	STANM=EWES,CPUID=A,SYMLN=42,STANO=08,APPLN=CXJA
	BSNCT	STANM=EWES,CPUID=A,SYMLN=44,STANO=08,APPLN=CXJA
*OUT	BSNCT	STANM=FAWN,CPUID=B,SYMLN=40,STANO=08,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=FAWN,CPUID=B,SYMLN=41,STANO=08,APPLN=CXJB
	BSNCT	STANM=FAWN,CPUID=B,SYMLN=42,STANO=08,APPLN=CXJB
	BSNCT	STANM=FAWN,CPUID=B,SYMLN=44,STANO=08,APPLN=CXJB
*OUT	BSNCT	STANM=FISH,CPUID=A,SYMLN=40,STANO=09,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=FISH,CPUID=A,SYMLN=41,STANO=09,APPLN=CXJA
	BSNCT	STANM=FISH,CPUID=A,SYMLN=42,STANO=09,APPLN=CXJA
	BSNCT	STANM=FISH,CPUID=A,SYMLN=44,STANO=09,APPLN=CXJA
*OUT	BSNCT	STANM=GOAT,CPUID=B,SYMLN=40,STANO=09,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=GOAT,CPUID=B,SYMLN=41,STANO=09,APPLN=CXJB
	BSNCT	STANM=GOAT,CPUID=B,SYMLN=42,STANO=09,APPLN=CXJB
	BSNCT	STANM=GOAT,CPUID=B,SYMLN=44,STANO=09,APPLN=CXJB
*OUT	BSNCT	STANM=HENS,CPUID=A,SYMLN=40,STANO=0A,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=HENS,CPUID=A,SYMLN=41,STANO=0A,APPLN=CXJA
	BSNCT	STANM=HENS,CPUID=A,SYMLN=42,STANO=0A,APPLN=CXJA
	BSNCT	STANM=HENS,CPUID=A,SYMLN=44,STANO=0A,APPLN=CXJA
*OUT	BSNCT	STANM=LAMB,CPUID=B,SYMLN=40,STANO=0A,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=LAMB,CPUID=B,SYMLN=41,STANO=0A,APPLN=CXJB
	BSNCT	STANM=LAMB,CPUID=B,SYMLN=42,STANO=0A,APPLN=CXJB
	BSNCT	STANM=LAMB,CPUID=B,SYMLN=44,STANO=0A,APPLN=CXJB
*OUT	BSNCT	STANM=LION,CPUID=A,SYMLN=40,STANO=0B,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=LION,CPUID=A,SYMLN=41,STANO=0B,APPLN=CXJA
	BSNCT	STANM=LION,CPUID=A,SYMLN=42,STANO=0B,APPLN=CXJA
	BSNCT	STANM=LION,CPUID=A,SYMLN=44,STANO=0B,APPLN=CXJA
*OUT	BSNCT	STANM=OWLS,CPUID=B,SYMLN=40,STANO=0B,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=OWLS,CPUID=B,SYMLN=41,STANO=0B,APPLN=CXJB
	BSNCT	STANM=OWLS,CPUID=B,SYMLN=42,STANO=0B,APPLN=CXJB
	BSNCT	STANM=OWLS,CPUID=B,SYMLN=44,STANO=0B,APPLN=CXJB
*OUT	BSNCT	STANM=RATS,CPUID=A,SYMLN=40,STANO=0C,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=RATS,CPUID=A,SYMLN=41,STANO=0C,APPLN=CXJA
	BSNCT	STANM=RATS,CPUID=A,SYMLN=42,STANO=0C,APPLN=CXJA
	BSNCT	STANM=RATS,CPUID=A,SYMLN=44,STANO=0C,APPLN=CXJA
*OUT	BSNCT	STANM=SEAL,CPUID=B,SYMLN=40,STANO=0C,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=SEAL,CPUID=B,SYMLN=41,STANO=0C,APPLN=CXJB
	BSNCT	STANM=SEAL,CPUID=B,SYMLN=42,STANO=0C,APPLN=CXJB
	BSNCT	STANM=SEAL,CPUID=B,SYMLN=44,STANO=0C,APPLN=CXJB
	BSNCT	STANM=STA1,CPUID=A,SYMLN=40,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA1,CPUID=A,SYMLN=41,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA1,CPUID=A,SYMLN=44,STANO=00,APPLN=CXJA
*OUT	BSNCT	STANM=STA2,CPUID=A,SYMLN=40,STANO=01,APPLN=CXJA FOR PAT Q
	BSNCT	STANM=STA2,CPUID=A,SYMLN=44,STANO=01,APPLN=CXJA
	BSNCT	STANM=STA3,CPUID=A,SYMLN=41,STANO=01,APPLN=CXJA
	BSNCT	STANM=STA4,CPUID=A,SYMLN=42,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA5,CPUID=A,SYMLN=42,STANO=01,APPLN=CXJA
	BSNCT	STANM=STA6,CPUID=A,SYMLN=43,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA7,CPUID=A,SYMLN=3E,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA8,CPUID=A,SYMLN=3F,STANO=00,APPLN=CXJA
	BSNCT	STANM=STA9,CPUID=A,SYMLN=45,STANO=00,APPLN=CXJA
	BSNCT	STANM=STB1,CPUID=B,SYMLN=40,STANO=00,APPLN=CXJB
	BSNCT	STANM=STB1,CPUID=B,SYMLN=41,STANO=00,APPLN=CXJB
	BSNCT	STANM=STB1,CPUID=B,SYMLN=44,STANO=00,APPLN=CXJB
*OUT	BSNCT	STANM=STB2,CPUID=B,SYMLN=40,STANO=01,APPLN=CXJBFOR PAT Q
	BSNCT	STANM=STB2,CPUID=B,SYMLN=44,STANO=01,APPLN=CXJB
	BSNCT	STANM=STB3,CPUID=B,SYMLN=41,STANO=01,APPLN=CXJB
	BSNCT	STANM=STB4,CPUID=B,SYMLN=42,STANO=00,APPLN=CXJB
	BSNCT	STANM=STB5,CPUID=B,SYMLN=42,STANO=01,APPLN=CXJB
	BSNCT	STANM=STB6,CPUID=B,SYMLN=43,STANO=00,APPLN=CXJB
	BSNCT	STANM=STB7,CPUID=B,SYMLN=3E,STANO=00,APPLN=CXJB

```

BSNCT      STANM=STB8,CPUID=B,SYMLN=3F,STANO=00,APPLN=CXJB
BSNCT      STANM=STB9,CPUID=B,SYMLN=45,STANO=00,APPLN=CXJB
BSNCT      STANM=STC1,CPUID=A,SYMLN=46,STANO=00,APPLN=CXJA
BSNCT      STANM=STC2,CPUID=A,SYMLN=47,STANO=00,APPLN=CXJA
BSNCT      STANM=STC3,CPUID=A,SYMLN=48,STANO=00,APPLN=CXJA
BSNCT      STANM=STC4,CPUID=A,SYMLN=49,STANO=00,APPLN=CXJA
BSNCT      STANM=STC5,CPUID=A,SYMLN=4A,STANO=00,APPLN=CXJA
BSNCT      STANM=STC6,CPUID=A,SYMLN=4B,STANO=00,APPLN=CXJA
BSNCT      STANM=STC7,CPUID=A,SYMLN=4C,STANO=00,APPLN=CXJA
BSNCT      STANM=STC8,CPUID=A,SYMLN=4D,STANO=00,APPLN=CXJA
BSNCT      STANM=STD1,CPUID=B,SYMLN=46,STANO=00,APPLN=CXJB
BSNCT      STANM=STD2,CPUID=B,SYMLN=47,STANO=00,APPLN=CXJB
BSNCT      STANM=STD3,CPUID=B,SYMLN=48,STANO=00,APPLN=CXJB
BSNCT      STANM=STD4,CPUID=B,SYMLN=49,STANO=00,APPLN=CXJB
BSNCT      STANM=STD5,CPUID=B,SYMLN=4A,STANO=00,APPLN=CXJB
BSNCT      STANM=STD6,CPUID=B,SYMLN=4B,STANO=00,APPLN=CXJB
BSNCT      STANM=STD7,CPUID=B,SYMLN=4C,STANO=00,APPLN=CXJB
BSNCT      STANM=STD8,CPUID=B,SYMLN=4D,STANO=00,APPLN=CXJB
EJECT 1
*
*
*
*****
*          CCPSTA - COMMUNICATION OPTIONS          *
*****
*
*      CCPSTA      KPUPD=120          COMUNICATION CNTL PGM KEYPT UPDATE
EJECT 1
*
*
*
*****
*          CCPOLL - CORE BLOCK SHUTDOWN AND RESTART          *
*****
*
*      CCPOL      AIDWN=300,          SYNCHRONOUS LINK SHUTDOWN LEVEL      X
*                  AIRES=150,          SYNCHRONOUS LINK RESTART LEVEL      X
*                  BSDWN=300,          BISYNC LINE SHUTDOWN LEVEL          X
*                  BSRES=150,          BISYNC LINE RESTART LEVEL            X
*                  L70DWN=300,        3270 LOCAL SHUTDOWN LEVEL            X
*                  L70RES=150        3270 LOCAL RESTART LEVEL
EJECT 1
*
*
*
*****
*          CCPERR - ERROR RECOVER FOR CCP          *
*****
*
*      CCPERR      BSCER=(10,7,7,16,255),  BISYNC LINE ERROR COUNTS      X
*                  CCPER=10,          UPDATE INTERVAL FOR ERROR CNTS X
*                  L70ER=(3,2),        3270 LOCAL: MAX RETRY/RESTART X
*                  PCERR=(20,3),        1052: MAX LINE ERR/MAX RETRIES X
*                  SLCER=(10,3,3,3,16)  SYNCHRONOUS LINK ERROR COUNTS
EJECT 1
*
*
*
*****
*          UIPROT - SYSTEM UTILITIES          *
*****
*
*      UTPROT      UTIL=CAP0,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY'
*      UTPROT      UTIL=CAP1,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY'
*      UTPROT      UTIL=CAP2,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY'
*      UTPROT      UTIL=CAP4,PGMNM=BXXX,UTCOM='TPF CAPTURE UTILITY'
*      UTPROT      UTIL=BUFC,PGMNM=CACP,UTCOM='TPF BUFC UTILITY'

```



```

EJECT 1
*
*
*
*****
*          SYNCLK - SYNCHRONOUS LINK CONTROL          *
*****
*
*          SYNCLK      (02,4D35,01,01,01,PSB),  PADDED SABRE LEVEL      X
*                    (02,4D35,01,01,02,PSB),  PADDED SABRE LEVEL      X
*                    (06,4D35,01,02,01,PSB),  PADDED SABRE LEVEL      X
*                    (06,4D35,01,02,02,PSB),  PADDED SABRE LEVEL      X
*          AILCB=36,      AI CXFRC/ENQ LCB-IN LEVEL IN DEC      X
*          AILST=READY,   ROUTING OF INPUT MSG BLKS CCP-OPZO      X
*          AIRTE=READY,   ROUTING OF INPUT MSG BLK CMR1-CMR3      X
*          ILCBQ=100,     NBR OF SLOTS IN INPUT LCB QUEUE        X
*          NLINK=3,       NBR OF SLC LINKS IN THE SYSTEM        X
*          NLRT=7,        NBR OF SLOTS IN SLC ROUTING TABLE     X
*          NPOOL=96,      NBR OF SLC INPUT POOL RECORDS         X
*          N1LNK=1,       NBR OF SLC LINKS W/1 AI LINE PAIR      X
*          OLCBQ=10,      NBR OF SLOTS IN OUTPUT LCB QUEUE       X
*          SLCEP=YES      INCLUDE SLC SUPPORT FOR 3705 IN EP
*
EJECT 1
*****
*          MSGRT - MESSAGE ROUTER SUPPORT          *
*****
*          Default value of MAXAP is (256-(m+S)), where      *
*          m is the number of system defined application,    *
*          which is equal to 2*(number of processor in the network)+1 *
*          (eg. we have user appl for proc A-E, so m=2*5+1=11. *
*          s is the number of optional system application,    *
*          which is 1 if RES0=YES, otherwise S=0.            *
*          Hey - Do it up!                                    *
*****
*          MSGRT COMEXIT=YES          MAX NBR OF ACTIVE USER APPLICATIONS
EJECT 1                                @000.000
*****
*          MSGRTA - MESSAGE ROUTER SUPPORT          *
*****
*
MSGRTA  APLIC=AAAA,APROC=A,ASNA=YES,MRECV=YES,APPL=(S,007),      X
        EDIT=DDL7,PERM=YES                                         @000.000
MSGRTA  APLIC=BBBB,APROC=A,ASNA=YES,EDIT=DDL7,APPL=(S,003)
MSGRTA  APLIC=CCCC,APROC=A,ASNA=YES,EDIT=DDL7,APPL=(S,003)
MSGRTA  APLIC=SSSS,APROC=*,EDIT=QIS1,SIGN=YES,                    X
        ASNA=YES                                                  @000.000
MSGRTA  APLIC=SSS1,APROC=*,EDIT=QIS1,SIGN=YES,                    X
        ASNA=NO                                                    @000.000
MSGRTA  APLIC=RES0,APROC=*,EDIT=UII1,APPL=P,                      X
        SIMFM=YES,SIGN=YES,RES=YES,USER=HPN                      @000.000
MSGRTA  APLIC=RBN1,APROC=B,EDIT=QIS1,SIGN=NO,                     X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                      @000.000
MSGRTA  APLIC=RCN1,APROC=C,EDIT=QIS1,SIGN=NO,                     X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                      @000.000
MSGRTA  APLIC=RDN1,APROC=D,EDIT=QIS1,SIGN=NO,                     X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                      @000.000
MSGRTA  APLIC=RZN1,APROC=Z,EDIT=QIS1,SIGN=NO,                     X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                      @000.000
MSGRTA  APLIC=RON1,APROC=0,EDIT=QIS1,SIGN=NO,                     X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                      @000.000
MSGRTA  APLIC=RAC1,EDIT=QIS1,APROC=A,ASNA=YES,MRECV=YES,         X
        RCPL=BASIC,USER=HPN                                       @000.000
MSGRTA  APLIC=RAC2,EDIT=QIS1,APROC=A,ASNA=YES,MRECV=YES,         X
        RCPL=BASIC,USER=HPN                                       @000.000
MSGRTA  APLIC=RBC1,EDIT=QIS1,APROC=B,ASNA=YES,MRECV=YES,         X
        RCPL=BASIC,USER=HPN                                       @000.000

```

```

MSGRTA  APLIC=RBC2,EDIT=QIS1,APROC=B,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RCC1,EDIT=QIS1,APROC=C,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RCC2,EDIT=QIS1,APROC=C,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RDC1,EDIT=QIS1,APROC=D,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RDC2,EDIT=QIS1,APROC=D,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=REC1,EDIT=QIS1,APROC=E,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=REC2,EDIT=QIS1,APROC=E,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RZC1,EDIT=QIS1,APROC=Z,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=RZC2,EDIT=QIS1,APROC=Z,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=R0C1,EDIT=QIS1,APROC=0,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
MSGRTA  APLIC=R0C2,EDIT=QIS1,APROC=0,ASNA=YES,MRECV=YES,      X
        RCPL=BASIC,USER=HPN                                @000.000
*
MSGRTA  APLIC=REN1,APROC=E,EDIT=QIS1,SIGN=NO,                  X
        ASNA=NO,RCPL=EXP,MRECV=NO,USER=HPN                  @000.000
*
*****
*   FOLLOWING IS FOR CICS LU_0 TESTING                           *
*****
MSGRTA  APLIC=ZZZZ,APROC=*,ASNA=YES,EDIT=QIS1,APPL=(S,018)
*
*****
*   FOLLOWING IS FOR C BENCHMARK APPLICATIONS                     *
*****
MSGRTA  APLIC=CBM1,APROC=*,ASNA=NO,EDIT=QCB1,                  X
        SIGN=NO,RCPL=BASIC,USER=HPN,MRECV=NO                @000.000
MSGRTA  APLIC=CBM2,APROC=*,ASNA=NO,EDIT=QCBF,                  X
        SIGN=NO,RCPL=BASIC,USER=HPN,MRECV=NO                @000.000
MSGRTA  APLIC=CBW1,APROC=*,ASNA=NO,EDIT=QCB1,                  X
        SIGN=NO,RCPL=BASIC,USER=WP1,MRECV=NO                @000.000
MSGRTA  APLIC=CBW2,APROC=*,ASNA=NO,EDIT=QCBF,                  X
        SIGN=NO,RCPL=BASIC,USER=WP1,MRECV=NO                @000.000
*****
*****
*   FOLLOWING IS FOR C APPLICATIONS                               *
*****
MSGRTA  APLIC=CSTA,APROC=*,ASNA=NO,EDIT=QCSA,                  X
        SIGN=NO,RCPL=BASIC,USER=HPN,MRECV=NO                @000.000
*****
*   FOLLOWING IS FOR MESSAGE ROUTER MULTI-POINT TESTING          *
*****
MSGRTA  APLIC=CXFT,APROC=*,EDIT=CXFT,SIGN=NO,                  X
        ASNA=YES                                              @000.000
*
*****
*   FOLLOWING IS FOR MESSAGE ROUTER TESTING ON CPU A              *
*****
MSGRTA  APLIC=CXJA,APROC=*,EDIT=CXJA                          @000.000
*
*****
*   FOLLOWING IS FOR MESSAGE ROUTER TESTING ON CPU B              *
*****
MSGRTA  APLIC=CXJB,APROC=*,EDIT=CXJA                          @000.000
*****
*   FOLLOWING IS FOR MESSAGE ROUTER TESTING ON CPU Z              *
*****
MSGRTA  APLIC=CXJZ,APROC=*,EDIT=CXJA                          @000.000

```

```

*
*****
*   FOLLOWING IS FOR BISYNC 3270 TESTING ON CPU B   *
*****
MSGRTA   APLIC=GAMB,APROC=*,EDIT=DDE6,SIGN=NO,      X
          ASNA=YES                                   @000.000
*
*****
*   FOLLOWING IS FOR 3270 MAP TESTING ON CPU A   *
*****
MSGRTA   APLIC=MSB1,APROC=A,EDIT=MSB1,SIGN=NO,      X
          USER=WP1                                   @000.000
*
*****
*   APPLICATIONS DEFINED AS SECONDARY LOGICAL UNITS (SLU'S) *
*****
*****
*   FOLLOWING IS FOR 3270 MAP TESTING ON CPU B   *
*****
MSGRTA   APLIC=MSB2,APROC=*,EDIT=MSB1,SIGN=NO,      X
          USER=WP1                                   @000.000
*
*****
*   FOLLOWING IS FOR TPF APPLICATION - CONFERENCE ROOM RESVNS *
*****
MSGRTA   APLIC=CRRS,APROC=*,EDIT=RSU1,SIGN=NO,RES=YES
*
*****
*   FOLLOWING IS FOR 3270 BSC TESTING - EXISTING APPLICATIONS *
*****
MSGRTA   APLIC=B320,APROC=*,EDIT=DDE1,SIGN=NO,      X
          ASNA=NO,RCPL=BASIC,MRECV=YES               @000.000
*
*****
*   THE FOLLOWING SEGMENTS ARE FOR GAMMA TESTING - *
*****
*****
*   BSS GAMMA APPLICATIONS *
*****
MSGRTA   APLIC=GMH1,APROC=E,EDIT=QIS1,SIGN=NO,      X
          ASNA=YES,RCPL=BASIC,MRECV=YES,USER=HPN     @000.000
MSGRTA   APLIC=GMH4,APROC=*,EDIT=QIS1,SIGN=NO,      X
          ASNA=NO,RCPL=BASIC,MRECV=NO,USER=HPN       @000.000
MSGRTA   APLIC=GMH5,APROC=B,EDIT=QIS1,SIGN=NO,      X
          ASNA=YES,RCPL=BASIC,MRECV=YES,USER=HPN,    X
          DEVTYPE=TYPE1                               @000.000
MSGRTA   APLIC=GMH6,APROC=B,EDIT=QIS1,SIGN=NO,      X
          ASNA=YES,RCPL=BASIC,MRECV=YES,USER=HPN,    X
          DEVTYPE=TYPE1                               @000.000
MSGRTA   APLIC=GMH7,APROC=B,EDIT=QIS1,SIGN=NO,      X
          ASNA=NO,RCPL=BASIC,MRECV=YES,USER=HPN,    X
          DEVTYPE=TYPE2                               @000.000
MSGRTA   APLIC=GMH8,APROC=B,EDIT=QIS1,SIGN=NO,      X
          ASNA=NO,RCPL=BASIC,MRECV=NO,USER=HPN,      X
          DEVTYPE=TYPE3                               @000.000
MSGRTA   APLIC=GMH9,APROC=C,EDIT=QIS1,SIGN=NO,      X
          ASNA=YES,RCPL=BASIC,MRECV=YES,USER=HPN,    X
          DEVTYPE=TYPE4                               @000.000
MSGRTA   APLIC=GMHA,APROC=C,EDIT=QIS1,SIGN=NO,      X
          ASNA=YES,RCPL=BASIC,MRECV=NO,USER=HPN,     X
          DEVTYPE=TYPE5                               @000.000
MSGRTA   APLIC=GMHB,APROC=C,EDIT=QIS1,SIGN=NO,      X
          ASNA=NO,RCPL=BASIC,MRECV=YES,USER=HPN,     X
          DEVTYPE=TYPE6                               @000.000
MSGRTA   APLIC=GMHC,APROC=C,EDIT=QIS1,SIGN=NO,      X

```

	ASNA=NO, RCPL=BASIC, MRECV=NO, USER=HPN, DEVTYPE=SHARED	X @000.000
MSGRTA	APLIC=GMHD, APROC=D, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE7	X X @000.000
MSGRTA	APLIC=GMHE, APROC=D, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=EXP, MRECV=NO, USER=HPN	X @000.000
MSGRTA	APLIC=GMHF, APROC=D, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=EXP, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMHG, APROC=D, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=EXP, MRECV=NO, USER=HPN	X @000.000
MSGRTA	APLIC=GMHH, APROC=E, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMHI, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=EXP, MRECV=NO, USER=HPN	X @000.000
MSGRTA	APLIC=GMHJ, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=EXP, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMHK, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=EXP, MRECV=NO, USER=HPN	X @000.000
MSGRTA	APLIC=GMHL, APROC=Z, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMHM, APROC=0, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMT1, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMT5, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=NO, USER=HPN	X @000.000
MSGRTA	APLIC=GMC3, APROC=C, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X cccccccc
MSGRTA	APLIC=GMA1, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE5	X
MSGRTA	APLIC=GMA2, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE6	X
MSGRTA	APLIC=GMA3, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE7	X
MSGRTA	APLIC=GMA4, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE1	X
MSGRTA	APLIC=GMA5, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN, DEVTYPE=TYPE2	X
MSGRTA	APLIC=GMA6, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMA7, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000
MSGRTA	APLIC=GMA8, APROC=A, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=HPN	X @000.000

\*

\*\*\*\*\*

\* WP GAMMA APPLICATIONS \*

\*\*\*\*\*

MSGRTA	APLIC=GMT2, APROC=E, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=EXP, MRECV=NO, USER=WP3	X @000.000
MSGRTA	APLIC=GMT3, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=WP2, DEVTYPE=TYPE3	X
MSGRTA	APLIC=GMT4, APROC=B, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=WP3, DEVTYPE=TYPE4	X
MSGRTA	APLIC=GMT6, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=EXP, MRECV=NO, USER=WP1	X @000.000
MSGRTA	APLIC=GMT7, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=YES, USER=WP2	X @000.000
MSGRTA	APLIC=GMT8, APROC=*, EDIT=QIS1, SIGN=NO, ASNA=NO, RCPL=BASIC, MRECV=NO, USER=WP3	X @000.000
MSGRTA	APLIC=GMC1, APROC=D, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=WP3, DEVTYPE=TYPE3	X
MSGRTA	APLIC=GMC4, APROC=Z, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=WP3	X @000.000
MSGRTA	APLIC=GMC5, APROC=0, EDIT=QIS1, SIGN=NO, ASNA=YES, RCPL=BASIC, MRECV=YES, USER=WP3	X @000.000

```

MSGRTA  APLIC=GMC2,APROC=C,EDIT=QIS1,SIGN=NO, X
        ASNA=YES,RCPL=BASIC,MRECV=YES,USER=WP3,DEVTYPE=TYPE4
*
*****
*      THE FOLLOWING SEGMENTS ARE FOR LU_0 APPL to APPL TESTING      *
*****
MSGRTA  APLIC=RAS1,EDIT=QIS1,APROC=A,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,018),SAWARE=YES 0000.000
MSGRTA  APLIC=RBS1,EDIT=QIS1,APROC=B,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,255),SAWARE=YES 0000.000
MSGRTA  APLIC=RCS1,EDIT=QIS1,APROC=C,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,255),SAWARE=YES 0000.000
MSGRTA  APLIC=RDS1,EDIT=QIS1,APROC=D,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,018) 0000.000
MSGRTA  APLIC=RES1,EDIT=QIS1,APROC=E,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,018) 0000.000
MSGRTA  APLIC=RZS1,EDIT=QIS1,APROC=Z,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,018) 0000.000
MSGRTA  APLIC=R0S1,EDIT=QIS1,APROC=0,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,018) 0000.000
MSGRTA  APLIC=DADA,EDIT=QIS1,APROC=A,ASNA=YES,APPL=(S,018)
MSGRTA  APLIC=EBEB,EDIT=QTD0,APROC=B,ASNA=LU62,APPL=(S,018), X
        USER=WP1 0000.000
MSGRTA  APLIC=FCFC,EDIT=QTD0,APROC=C,ASNA=LU62,APPL=(S,018), X
        USER=WP1 0000.000
MSGRTA  APLIC=GDGD,EDIT=QTD0,APROC=D,ASNA=LU62,APPL=(S,018), X
        USER=WP1 0000.000
MSGRTA  APLIC=HEHE,EDIT=QTD0,APROC=E,ASNA=LU62,APPL=(S,018), X
        USER=WP1 0000.000
MSGRTA  APLIC=IZIZ,EDIT=QTD0,APROC=Z,ASNA=LU62,APPL=P, X
        USER=WP1 0000.000
MSGRTA  APLIC=J0J0,EDIT=QTD0,APROC=0,ASNA=LU62,APPL=P, X
        USER=WP1 0000.000
MSGRTA  APLIC=RANU,EDIT=QNU3,APROC=A,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,255),SAWARE=YES 0000.000
MSGRTA  APLIC=RBNU,EDIT=QNU3,APROC=B,ASNA=YES,MRECV=YES, X
        RCPL=BASIC,USER=HPN,APPL=(S,255),SAWARE=YES 0000.000
*
*****
*      THE FOLLOWING SEGMENTS ARE FOR LU62 APPLICATIONS      *
*****
MSGRTA  APLIC=LLLL,EDIT=CHDD,APROC=A,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=PPPP,EDIT=CHDD,APROC=B,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=XXXX,EDIT=CHDD,APROC=C,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=YYYY,EDIT=CHDD,APROC=D,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=WWW,EDIT=CHDD,APROC=E,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=DDDD,EDIT=CHDD,APROC=Z,ASNA=LU62,APPL=P 0000.000
MSGRTA  APLIC=0000,EDIT=CHDD,APROC=0,ASNA=LU62,APPL=P 0000.000
MSGRTA  APLIC=LALA,EDIT=CHDD,APROC=A,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=PBPB,EDIT=CHDD,APROC=B,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=XCXC,EDIT=CHDD,APROC=C,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=YDYD,EDIT=CHDD,APROC=D,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=WEWE,EDIT=CHDD,APROC=E,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=AZAZ,EDIT=CHDD,APROC=Z,ASNA=LU62,APPL=P 0000.000
MSGRTA  APLIC=Y0Y0,EDIT=CHDD,APROC=0,ASNA=LU62,APPL=P 0000.000
MSGRTA  APLIC=L6G2,EDIT=CHDD,APROC=*,ASNA=LU62,APPL=P, X
        USER=WP1 0000.000
MSGRTA  APLIC=L6G5,EDIT=CHDD,APROC=*,ASNA=LU62,APPL=(S,018), X
        USER=WP2 0000.000
MSGRTA  APLIC=L621,EDIT=CHDD,APROC=*,ASNA=LU62,APPL=(S,018)
*
*****
*      THE FOLLOWING SEGMENT IS FOR WRAP TEST FUNCTION ACTIVATION  *
*****
MSGRTA  APLIC=WRAP,EDIT=QTD6,APROC=*,ASNA=YES 0000.000
*
*****

```

```

*      TPF's Control Point (CP) Application Name      *
*****
MSGRTA  APLIC=CP41,EDIT=CHDD,APROC=*,ASNA=LOCP      @000.000
*****
*      THE FOLLOWING SEGMENTS ADDED FOR LU62 TESTING  *
*****
MSGRTA  APLIC=APPA,EDIT=CHDD,APROC=A,ASNA=LU62      @000.000
MSGRTA  APLIC=APPC,EDIT=CHDD,APROC=*,ASNA=APPC      @000.000
MSGRTA  APLIC=LUP0,EDIT=CHDD,APROC=*,ASNA=LU62      @000.000
MSGRTA  APLIC=LUP1,EDIT=CHDD,APROC=*,ASNA=LU62      @000.000
*
*****
*      THE FOLLOWING SEGMENTS ARE FOR CTC LU6.2 SESSIONS  *
*****
MSGRTA  APLIC=CTCA,EDIT=CHDD,APROC=A,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=CTCB,EDIT=CHDD,APROC=B,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=CTCC,EDIT=CHDD,APROC=C,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=CTCD,EDIT=CHDD,APROC=D,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=CTCE,EDIT=CHDD,APROC=E,ASNA=LU62,APPL=(S,018)
MSGRTA  APLIC=CTCZ,EDIT=CHDD,APROC=Z,ASNA=LU62,APPL=P @000.000
MSGRTA  APLIC=CTC0,EDIT=CHDD,APROC=0,ASNA=LU62,APPL=P @000.000
MSGRTA  APLIC=SVCB,EDIT=CHDD,APROC=B,ASNA=APPC      @000.000
MSGRTA  APLIC=SVCC,EDIT=CHDD,APROC=C,ASNA=APPC      @000.000
MSGRTA  APLIC=SVCD,EDIT=CHDD,APROC=D,ASNA=APPC      @000.000
MSGRTA  APLIC=SVCE,EDIT=CHDD,APROC=E,ASNA=APPC      @000.000
MSGRTA  APLIC=SV CZ,EDIT=CHDD,APROC=Z,ASNA=APPC      @000.000
MSGRTA  APLIC=SV C0,EDIT=CHDD,APROC=0,ASNA=APPC      @000.000
*
*****
*      THE FOLLOWING SEGMENTS ARE FOR NON SNA SLC TESTING  *
*****
MSGRTA  APLIC=SLCA,EDIT=SLCA,APROC=*                @000.000
MSGRTA  APLIC=TSIM,EDIT=QKA8,APROC=*,SIGN=NO,        X
        ASNA=YES,RCPL=BASIC,MRECV=NO                @000.000
EJECT 1                                              @000.000
*
*
*
*
*****
*      DATACO - DATA COLLECTION PARAMETERS              *
*****
*
DATACO  CIT=35,          MAX NBR OF CITIES ASSOCIATED W/IA  X
        IAC=30,          MAX NBR OF INTERCHANGE ADDRS/CITY  X
        IAS=200,         MAX NBR OF UNIQUE LN IA'S          X
        INTVL=300,       MAX NBR OF SAMPLING INTERVALS/RUN  X
        NPGM=35,         NBR OF PGM NAMES PLACED IN 'SM' RCD X
        PGMS=600,        NBR OF PROGRAMS REPORTED           X
        RECID=100,       MAX NBR OF UNIQUE POOL & RECORD IDS X
        TMC=700,         MAX NBR OF TERMINALS/CITY          X
        USERN='TPF 4.1 HPO SYSTEM', HOW MANY 1/16th BENDS?  X
        WT=5,            WEIGHTING FACTOR FOR LOW SPEED MSGS X
        TNODS=25000,     NUMBER LOCAL & CROSS DOMAIN NODES X
        WT2=3,           WEIGHTING FACTOR FOR ROUTER MSGS
EJECT 1
*
*
*
*****
*      RESCAP - CAPTURE RESTORE                          *
*****
*
RESCAP  ABRT0=30,        MAX TIME ALLOWED ENTRIES TO ABORT  X
        CAPTO=6144,      PAUSE TIME OUT FACTOR IN SECONDS   X
        DKERR=(0,0),     DISK ERROR COUNTS                  X
        SECUR=60,        INTERVAL IN SECS FOR SECURITY CHECK X

```

```

                                TPERR=(3,60,3),    TAPE ERROR COUNTS      X
                                TPREW=76           MAX REWIND/UNLOAD TIME IN SECONDS
EJECT 1
*
*
*
*****
*          LOGCAP - TAPE CAPTURE PARAMETERS          *
*****
*
    LOGCAP    RTU=(DEVA,0,DEVB,2),  DEVICE TYPE AND RELATIVE MOD NBR  X
              STAMP=01              INTERVAL IN MINUTES FOR TIME STAMP
EJECT 1
*
*
*
*****
*          DDCCAP - DASD CAPTURE RESTORE PARAMETERS    *
*****
*
    DDCCAP    KPUPA=25,KPUPB=25,INTAP=0-3,INTBP=0-3      @000.000
EJECT 1
*
*
*
*****
*          INDSN - ALLOCATOR DECK FRO THIS SYSTEM      *
*****
*
    INDSN      SALMEM=USRTPF,                                X
               RIATDSN=ACP.RIAT.BSS40,                      X
               DFMAC1=ACP.MACR01.PUT40DF,                    X
               DFHDR1=ACP.CHDR.PUT40DF,                       X
               DFSRC1=ACP.SRCE.RT1.PUT40DF,                   X
               DFSRC2=ACP.SRCE.RT1.PUT40DF,                   X
               DFCSR1=ACP.CSRCE.RT.PUT40DF,                   X
               DFOBJ1=ACP.OBJ.PUT40DF.BSS,                    X
               DFLNK1=ACP.LINK.PUT40DF.BSS                     @000.000
*
*
*****
*          GENSIP - SIP GENERATION PARAMETERS FOR THIS RUN  *
*****
*
    GENSIP      ANTPDS=ACP.ANTS.REL40.SQ,      OFFLINE SNA TABLE GEN X
               LGFDV=(3390,75),    DEVICE TYPE OF GENERAL FILE      X
               VOLNLF=0999,        VOLUME SERIAL NBR OF GENERAL FILE  X
               OUTCL=(A,A,A),      PRINT OUTPUT CLASSES (CP,RT,OL)    X
               PLILK=PLI.PLIBASE,   X
               PLISR=(PLI.SIBMBASE,PLI.PLIBASE), X
               USPROC=SYS2.PROCLIB, SET UP USER PROC LIBRARY        X
               RELNN=40,           RELEASE LEVEL                      X
               LISTAPE=SYSOUT,     ASSEMBLIES TO PRINTER              X
               ASMAIL=YES,        ASSEMBLE ALL                       X
               LINKALL=YES,       LINK EVERYTHING                    X
               ASMTYPE=I,         Use high-level assembler           X
               XREF=(NO,FULL,SHORT), CROSS REFERENCE ON ASSEMBLIES    X
               REGN=8192,         REGION SIZE TO BE USED BY ASSEMBLER X
               CLIBPFX=SYS1.CEE,   X
               CLNGPFX=SYS1.CBC,   X
               CCOMPTP=G,          X
               CRUNTIM=LE          @000.000
END ,

```

---

## Sample SIP Stage I Output - SIP Report

The sample SIP stage I output report was deleted. You can run SIP stage I to get this report. See “Sample SIP Stage I Input” on page 351 for more information.



## Appendix C. SIP Stage II Job Summary

The following job summary provides you with a guide to SIP Stage II job execution. In the case of hardware errors or other non-SIP problems (for example, space allocation problems) it shows you what must be reexecuted if a particular job or step fails. It identifies each job and step and any associated jobs or steps that must be successfully executed before a particular job or step is run. The function for each job or step is also identified.

**Note:** Most jobs depend on successful completion of the first job, SIPF2A, step F2A1U, which updates the macro library.

This appendix reflects a full Stage I generation only. The subsystem generation capability of the MDBF environment provides an added degree of selectability that is not reflected in this appendix. Also, most jobs require that you run the FACE Table Generator (FCTBG) before any of the Stage II jobs are run because the FCTBG creates parts of SYCON and SYSEQC, and these macros are assembled as parts of many segments.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPF2A - Updates macro And SIP Support Libraries					
	F2A1U	Successful completion of this step is a requirement for almost all of the other jobs.	none	-	macro updates: LINEQ SYCON SYMSEQ SSSET CK9KC SDAT SYSETK SYGLBK SYSET SYGLB CC0CC CC1CC BSSSET BXAXF UIPROT C\$GTSZ
	F2A2U		none	-	updates procedures used for MASM assemblies
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPG1A - Library Support Programs					
	G1A1A		SIPF	all	MASM assembly
	G1A2A		SIPF	all	DRVA assembly
	G1A3L		SIPG1A	G1A1A G1A2A	Link edit of: MASM DRVA
	G1A4A		SIPF	all	Assembly of: DCRS DCR2 DREF JULTOACT
SIPG1B					
	G1B1B	FCTBG must be run	SIPF	all	Assembly of:

first so that DFAD  
will have access  
to STCEQ and SYSEQC.

VCRS  
DFAD  
FMTR  
ASMP  
CCLOAD  
CCDUMP  
BKDR  
BKDIND  
PRSR  
SADPRT  
Link edits of:

G1B6L

SIPG1A G1A4A  
SIPG1B G1B1B

VCRS  
DCRS  
DCR2  
DREF  
JULTOACT  
FMTR  
ASMP  
CCLOAD  
CCDUMP  
BKDR  
BKDIND  
PRSR  
SADPRT  
Link edit of:

G1B7L

SIPG1B G1B1B

DFAD

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPG2A - Loaders, Linkers, IPL Program and System Test Compiler					
	G2A1A		SIPF	all	Assembly of:
			SIPF	all	STCC
			SIPF	all	STCI
			SIPF	all	STCL
			SIPF	all	SDMU
		FCTBG should be run first to produce STCEQ required by DGR0	SIPF	all	DGR0
	* - these segments are concerned with mapping support and will not appear unless mapping is generated.				ASL0L *
					ASLPR *
					ASLOM *
SIPG2B					
	G2B1A				OSTGRT
					OSTGIP
					OSTGUP
					OSTGOT
					OSTGP2
					TLDR
			SIPF	all	IPLA
					IPLB
					LEDT
	G2B3L	name = EDITOR	SIPG2B	G2B1A	Link edit of
					ASL0L
	G2B4L	name = STC	SIPG2A	G2A1A	Link edit of
			SIPG2B	G2B1A	STC
	G2B6L		SIPG1A	G2A4A	Link edit of
			SIPG2B	G2B1A	OSTGvv
	* - Job step SIPG2B will not link edit OSTG if ACF support is not generated.				
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPG4A - TPF Maintenance Support System					
	G4A1A	CBQ offline	SIPF	all	Assembly of:

		segments are only included in a MPIF environment.			AMX1, HLST, CBQPRT, CBQ4, CBQ5, and CBQ6
	G4A1L		SIPG4A	G4A1A	Link edits of: AMX1, HLST, CBQPRT, CBQ4, CBQ5, and CBQ6
SIPG5A - File And DASD Support					
	G5A1U		None		Source update for CVZD
	G5A2A		SIPF	all	Assembly of
	G5A3B	FCTBG should be run first so that RIAT will know about pool types coded.	SIPG5A	G5A1U	CVZD
			SIPF	all	Assembly of RIAT
	G5A4A		SIPF	all	Assembly of: BPR0, BRFA
	G5A5C		SIPF	all	Compile of DY00
	G5A6C		SIPF	all	Compile of DY04
	G5A7C		SIPF	all	Compile of DY06
	G5A9L		SIPG5A	G5A4A	Link edit of BPR0
		includes CVZD & BRFA	SIPG5A	G5A2A	Link edit of BRFA
	G5A10L	includes DY0 programs plus CVZD	SIPG5A	G5A2A G5A4A G5A5C thru G5A7C	Link edit of DYOPM
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPG6A -	Data Reduction				
	G6A1U		none		Source update for JPC0
	G6A2C thru G6A10C		SIPF SIPG6A	all G6A1U	Compile of: JRA0 - JRA6, JRF1, JRF4
SIPG6B	G6B1C thru G6B10C		SIPF SIPG6A	all G6A1U	Compiles of: JRF5, JRF6, JRM1, JRM2, JRM4, JRM5, JRP1, JRP3, JRS1, JRS3
	G6B11L		SIPG6A SIPG6B	all all	Link edit of DATAREAD.
SIPG6D	G6D1A G6D1B		SIPF SIPG6D	all G6D1A	Assem CHQI Link CHQI
SIPG6E -	DB2 PRECOMPILER (TPF PHASE)				
	G6E1A		none		Compile and link edit of DB2PP.
SIPG6F -	PIUPRT Create				
	G6F1A		none		Compile and link edit of PIUPRT
SIPG6G -	OLDR Create				
	G6G1A		none		Compile OLDR
	G6G1B		none		Compile COLR

G6G1C	none	Compile UELR
SIPG6H - OLDR Create		
G6H1A	none	Compile TOKN
G6H1B	none	Compile CBLD
G6H1C	none	Compile STUB
G6H1D	none	Compile RAISST
G6H1E	SIPG6H all	Link of CBLD
G6H1F	SIPG6H all	Link of STUB
* - Job steps G6H1G and G6H1H will not be created for a sub-system.		
G6H1G	none	Compile LIBI
G6H1H	SIPG6H all	Link of LIBI
SIPG6I - CMQMPP Create		
MQTR	none	Compile and link edit of CMQMPP
SIPG6J - IPTPRT (IP Trace Report Generator)		
G6J1A	none	Compile and link-edit of IPTPRT

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
-------------	--------------	----------	-----------	---------------	----------

SIPHn - Generates the UFOF assemble and link edit JCL for the TPFDF product

H1nA	SIPF	all	Assembly of UFOF
	H1nB		Link edit of UFOF

n = Job number (1 - 999).

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
-------------	--------------	----------	-----------	---------------	----------

Control Program and System Loader for General File FCTB Link Edit and System test postprocessor creation.

SIP11A	I1A1A	SIPF SIPG1A	all G1A1A G1A2A G1A3L	Multiple assembly of control program CSECTs
SIP11B	I1B1A	SIP11A	I1A1A	Multiple assembly of control programs
SIP11C	I1C1A	SIP11B	I1B1A	Multiple assembly of control programs
SIP11D	I1D1A	SIP11C	I1C1A	Multiple assembly of control programs
SIP11E	I1E1A	SIP11D	I1D1A	Multiple assembly of control programs
SIP11F	I1F1A	SIP11E	I1E1A	Multiple assembly of control programs
SIP11G	I1G1A	SIP11F	I1F1A	Multiple assembly of control programs
SIP11H	I1H1A	SIP11G	I1G1A	Multiple assembly of control programs
SIP11I	I1I1A	SIP11H	I1H1A	Multiple assembly of control programs
SIP11J	I1J1A	SIP11I	I1I1A	Multiple assembly of control programs
SIP11K	I1K1A	SIP11K	I1K1A	Multiple assembly of control programs
SIP11L	I1L1A	SIP11L	I1L1A	Multiple assembly of control programs
SIP11M	I1M1A	SIP11M	I1M1A	Multiple assembly

of control programs					
SIPI2E	I2E5L	Creates control program load module (CPS0)	SIPI1	all	Link edit of CP CSECTs
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPI3A	I3A1A		SIPF	all	Assemble STPP
SIPI3B	I3B1A		SIPF	all	Assemble PPCP, BMP0-7, BMGL, LTPP, LTPQ, ACPL, ICDF
	I3B1B		none	none	Assemble USR1, USR2
	I3B3L		SIPI	CCMCDC assem	Link Edit PPCP
	I3B4L		SIPI3B	I3B1A	Link Edit BMGL
	I3B5L		SIPI3B	I3B1A	Link Edit LTPP
	I3B6L		SIPI3B	I3B1A	Link Edit LTPQ
	I3B7L	FCTBG must be run first	SIPF	all	Assemble CONK, CTSD
	I3B8L		SIPI3B	I3B7L	Link Edit FCTB
SIPKISGT	- punch out SYNLST copy member and SIGT program for Global Synchronization				
	KSIGT1U		SIPF1A		Source update for SIGTvv (Global Sync. input deck)
SIPKSYNC	KSYNC1U		SIPF1A		Source update for SYNLST copy member
SIPK1A	K1A1A		SIPKISGT	all	Assembly of SIGTvv
			SIPKSYNC	all	
SIPSAL1A	- Creation and Execution of System Allocator				
	SAL1A2D		none	none	Add C\$IDFUNC (Function switch settings)
	SAL1A2E		SIPSAL1A	SAL1A2D	Compile, link and run SALO
	SAL1A2G		SIPSAL1A	SAL1A2E	Assemble IPAT
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPJAn	- Multiple Compilation of C functions for C load module				
	J4Anm		SIPF	all	Multiple compilation of C functions
			SIPG1A	G1A1A	
				G1A2A	
				G1A3L	
	n = Job step number. The list of C functions in SPPGML (ISC type) is compiled (based on dependency function switch) ten at a time.				
	m = Job execute step number. Minimum of one execute step per job.				
SIPJBn	- Multiple assembly of functions for C and C++ load modules				
	J4Bnm		SIPF	all	Multiple assembly of functions
			SIPG1A	G1A1A	
				G1A2A	
				G1A3L	
	n = Job step number. The list of assembler functions				

in SPPGML (ISA type) is assembled  
(based on dependency function switch)  
ten at a time.

m = Job execute step number. Minimum of one execute  
step per job.

SIPJCn - Multiple assembly of functions for C++ load modules  
J4Cnm SIPF all Multiple  
SIPG1A G1A1A assembly  
G1A2A of  
G1A3L functions  
n = Job step number. The list of C++ functions  
in SPPGML (CPP type) is compiled  
(based on dependency function switch)  
one at a time.  
m = Job execute step number. Minimum of one execute  
step per job.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
----------------	-----------------	----------	--------------	------------------	----------

SIPJDn - Multiple Compilation of C functions for the C load module of  
the TPFDF product  
J4Dnm SIPF all Multiple  
SIPG1A G1A1A compilation  
G1A2A of C  
G1A3L functions  
n = Job step number. The list of C functions  
in SPPGML (ISC type) is compiled  
(based on dependency function switch)  
ten at a time.  
m = Job execute step number. Minimum of one execute  
step per job.

SIPJEn - Multiple assembly of functions for the C load module of the  
TPFDF product  
J4Enm SIPF all Multiple  
SIPG1A G1A1A assembly  
G1A2A of  
G1A3L functions  
n = Job step number. The list of assembler functions  
in SPPGML (ISA type) is assembled  
(based on dependency function switch)  
ten at a time.  
m = Job execute step number. Minimum of one execute  
step per job.

SIPJFn - Multiple Compilation of C++ functions for the C load module of  
the TPFDF product  
J4Fnm SIPF all Multiple  
SIPG1A G1A1A compilation  
G1A2A of C++  
G1A3L functions  
n = Job step number. The list of C++ functions  
in SPPGML (CPP type) is compiled  
(based on dependency function switch)  
ten at a time.  
m = Job execute step number. Minimum of one execute  
step per job.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
----------------	-----------------	----------	--------------	------------------	----------

SIPJ1A - General File Keypoints And Message Router Tables  
J1A1U SIPF all Source update  
for CTKAGF  
and CTKVGF  
J1A1E SIPF all Compile, link  
SIPSAL1A SAL1A2G and run

			SIPJ1A	J1A1U	Assembly of CTKAGF, CTKVGF Assemble ALDR and TLDR
	J1A2A		SIPG5A	G5A3B	
	J1A3A		SIPG2B	G2B1A	
	J1A4A		SIPK1A	K1A1A	
	J1A4B		SIPJ1A	J1A1U	Compile CILI
	J1A4C		SIPF	all	Compile TLDRMN
	J1A4D		SIPF	all	Compile NLDLT
	J1A5A		SIPF	all	Compile NLDTRL
			SIPJ1A	J1A3A J1A4A J1A4B J1A4C J1A4D	Link-edit of the TPFLDR (ALDR, TLDR, and OLDR)
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPJ1B	J1B1U	RCIT program data records	none		Source update for COHA, COHB COHC ... (RCAT initialization table).
	J1B2A		SIPF SIPJ1B	all J1B1U	Assembly of COHA, COHB, COHC...
	J1B4U	Source code placed onto library specified by ANTPDS key-word of GENSIP macro.	SIPJ1B	J1B1U J1B2A	Source update OSTG ANTPDS definitions
SIPJ2A - Real Time Keypoints					
	J2A1U		SIPPAL1A SIPJ1A	SAL1A1U J1A1E	Source updates for real time keypoints:
	J2A2A		SIPF SIPJ11 SIPJ2A	all all J2A1U	MASM assembly real time keypoints.
SIPJ3A - Communication Keypoints, Tape Label Records And UII Table					
	J3A1U		none		Source updates for CTKX,COSY, CCKP
	J3A2A		SIPF SIPJ3A	all J3A1U	MASM assembly of: CTKP,BXAX, COSY,CCKP
	J3A2U	BSNCT table records	SIPJ1B	J1B1U	Source update for CRS1, CRS2 CRS3, ....
	J3A4A		SIPF SIPJ3A	all J3A3U	MASM assembly of CRS1, CRS2, CRS3, ....
	J3A5U	BBSAT table records	none		Source update for CRS0.
	J3A6A		SIPF SIPJ3A	all J3A5U	Assembly of CRS0
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function

SIPJ4n - Multiple Assembly of Real-Time Programs  
 J4Anm                               SIPF   all           Multiple  
                                      SIPG1A G1A1A       assembly of  
    G1A2A       real-time  
    G1A3L       programs  
 n = Job step number. The list of real-time programs  
     in SPPGML (and SPIPML for a WTC user) is  
     assembled (based on dependency function switch)  
     ten at a time.  
 m = Job execute step number. Minimum of one execute  
     step per job.

SIPJ5n - Multiple Assembly of WTC Offline Components.

J5N                               SIPF   all           Multiple  
                                      SIPG1A G1A1A       assembly  
    G1A2A       of WTC  
    G1A3L       offline  
   components  
 n = Job and step number for the WTC offline  
     components.

SIPJ6n - Link edit of WTC offline components.  
 J6n                               SIPJ5n J5n  
 n = Job and step number for the WTC offline components.

SIPJ7n - Multiple Assembly of User Real-Time Programs  
 J7Anm                               SIPF   all           Multiple  
   assembly  
   of user  
   real-time  
   programs  
 n = Job number. The list of user real-time programs  
     in SPPGML is assembled ten programs at a time.  
 m = Job execute step number. Minimum of one execute  
     step per job.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
----------------	-----------------	----------	--------------	------------------	----------

SIPJ8n - Multiple Compilation of C Real-Time Programs  
 J8Anm                               SIPF1A all           Multiple  
                                      SIPF1B all           Compilation  
                                      SIPF2A F2A2U       of TPF C  
                                      SIPG1A G1A1A       real-time  
    G1A2A       programs  
    G1A3L

\*

n = Job number. The list of TPF real-time programs  
     in SPPGML is compiled ten programs at a time.  
 m = Job execute step number. Minimum of one execute  
     step per job.  
 \* = Additionally, the #pragma statements must be  
     available from the assembly of C000.

SIPJ9n - Multiple Compilation of User C Real-Time Programs  
 J9Anm                               SIPF1A all           Multiple  
                                      SIPF1B all           Compilation  
                                      SIPF2A F2A2U       of User C  
                                      SIPG1A G1A1A       real-time  
    G1A2A       programs  
    G1A3L

\*

n = Job number. The list of TPF real-time programs



in SPPGML is compiled ten programs at a time.

m = Job execute step number. Minimum of one execute step per job.

\* = Additionally, the #pragma statements must be available from the assembly of C000.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPJ10n		- Multiple Assembly of Real-Time Programs for the TPFDF product			
	J10Anm		SIPF	all	Multiple
			SIPG1A	G1A1A	assembly of
				G1A2A	real-time
				G1A3L	programs
	n	= Job step number. The list of real-time programs in SPPGML (and SPIPML for a WTC user) is assembled (based on dependency function switch) ten at a time.			
	m	= Job execute step number. Minimum of one execute step per job.			
SIPL1A		- Initialize and Format Loader General File DASD			
	L1A1E		SIPG2B	G2B1A	Initialize Loader General File
	L1A2E		SIPG1B	G1B6L	Format Loader
	L1A3E		SIPG1B	G1B6L	Format Loader
Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPL2A		- Update And Assemble Release PARS List			
	L2A1U		SIPG5	G5A3A*	Source update
			SIPJ3	all*	for: PARSvv
			SIPJ4	all*	(Release PARS List)
	IF TPFDF THEN				
	L2B1U		SIPG5	G5A3A*	Source update
			SIPJ3	all*	for: PARSyy
			SIPJ4	all*	(Release PARS List)
	IF NOT WTC THEN				
	L2A3A	Current release PARS list	SIPL2A	L2A1U	PARSvv assembly
	ELSE	/* user requested WTC code */			
	L2B1U		SIPG5	G5A3A	Source update
			SIPJ3	all*	for: PARSww
			SIPJ4	all*	(IPARS
			SIPJ5	all*	PARS List if
					USER = WTC in
					CONFIG Macro)
	ENDIF				
	L2B3A	Current release PARS list, and WTC PARS list if USER=WTC	SIPL2A	L2A1U	PARSvv assembly
	vv	= Version number of current release.			
	ww	= WTC version number of WTC PARSLIST.			
	yy	= TPFDF Version number of current release.			

#### Notes:

1. Jobs SIPG5 (step G5A3A) and Jobs SIPJ3 and SIPJ4 are previous dependencies only in the sense that version numbers must agree and all programs added to the Release PARS List must be assembled (available for loading).

2. PARS List will include all real-time programs included in SPPGML and SPIPML for a WTC user (except those excluded by functional dependency switch). In addition, programs BXAX, COSY, CCKP, and COHA-COHG are excluded from the PARS list because of the necessity of only loading these programs once.

SIPL3A - Update the Load Deck

L3A1U	ALL*	-	Source update for: LOADDECK
-------	------	---	--------------------------------

**Note:** All previous jobs are considered previous dependencies in the sense that version numbers must agree and, for execution of the LOADDECK (ALDR), all of Stage II must be complete.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
SIPL3B - List the Load Deck	L3B2P		SIPL3A	L3A1U	Print the LOADDECK

SIPL4A - Create, Update And List Online DASD Formatter Decks

L4A1U	none	none	Creates and provides source update containing FMTRDECK
-------	------	------	--

FCTBG must be run  
before this job to  
create the necessary  
FMT cards.

SIPL4B	L4B1P		SIPL4A	L4A1U	Prints the FMTRDECK
--------	-------	--	--------	-------	------------------------

**Note:** FMTRDECK will contain the JCL necessary to initialize and format the first module of each device type specified by the ONLFIL macros. You only need to change the VOLSER parameter of the JCL to format the remaining modules of each device type.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
----------------	-----------------	----------	--------------	------------------	----------

\* Job SIPL51 will not be created for subsystems.

SIPL51 - Execute the Library Interface Tool for libraries, including the TPFDF library.

L511A	SIPG6H	G6H1H	Execute LIBI for CTAL
L511B	SIPG6H	G6H1H	Execute LIBI for CISO
L511C	SIPG6H	G6H1H	Execute LIBI for CTBX
L511D	SIPG6H	G6H1H	Execute LIBI for COMX
L511E	SIPG6H	G6H1H	Execute LIBI for CJ00
L511F	SIPG6H	G6H1H	Execute LIBI for CMQI
L511G	SIPG6H	G6H1H	Execute LIBI for CTHD
L511H	SIPG6H	G6H1H	Execute LIBI for CTDF

# S IPL52 - Execute the DLM Stub Generator

L521A	SIPG6H	G6H1F	Execute STUB
-------	--------	-------	--------------

**Note:** Jobs that prelink and link libraries will not be created for subsystems.

Current Job	Current Step	Comments	Prior Job	Prior Step(s)	Function
* SIPL5n - Multiple libraries and C load module prelink and link-edit jobs					
	L5nA		SIPG6H	G6H1E	Execute the Build tool to produce INCLUDE deck
	L5nB		SIPJA##	all	Prelink and link program
			SIPJB##	all	
			SIPL51	all	
			SIPL52	all	

n = Job number (5 - 999). The list of C load module E-type programs in SPPGML (type ICL) are prelinked and link-edited.

SIPL6n - Multiple C load module prelink and link-edit jobs for the TPFDF product					
	L6n		SIPG6H	G6H1E	Execute the Build tool to produce INCLUDE deck
					Prelink and link program

n = Job number (1 - 999). The list of C load module E-type programs in SPPGML (type ICL) are prelinked and link-edited.



---

## Appendix D. Sample Central Site Configurations

The following configurations are samples to aid in generating a TPF system. They were chosen to illustrate some design choices in TPF configurations. The SIP input (but only that which is required for the hardware depicted) is shown for each.

The two configurations that are shown and described are:

- A base only system, illustrated by Figure 30 on page 486. It represents a minimum TPF system without the HPO feature. It also shows a configuration with a backup processor and the necessary switching capability to the second processor for the purpose of switchover.
- A loosely-coupled system, illustrated by Figure 31 on page 490. The loosely-coupled facility is part of the HPO feature. The illustration depicts a system with two active processors and one backup processor in the LC complex.

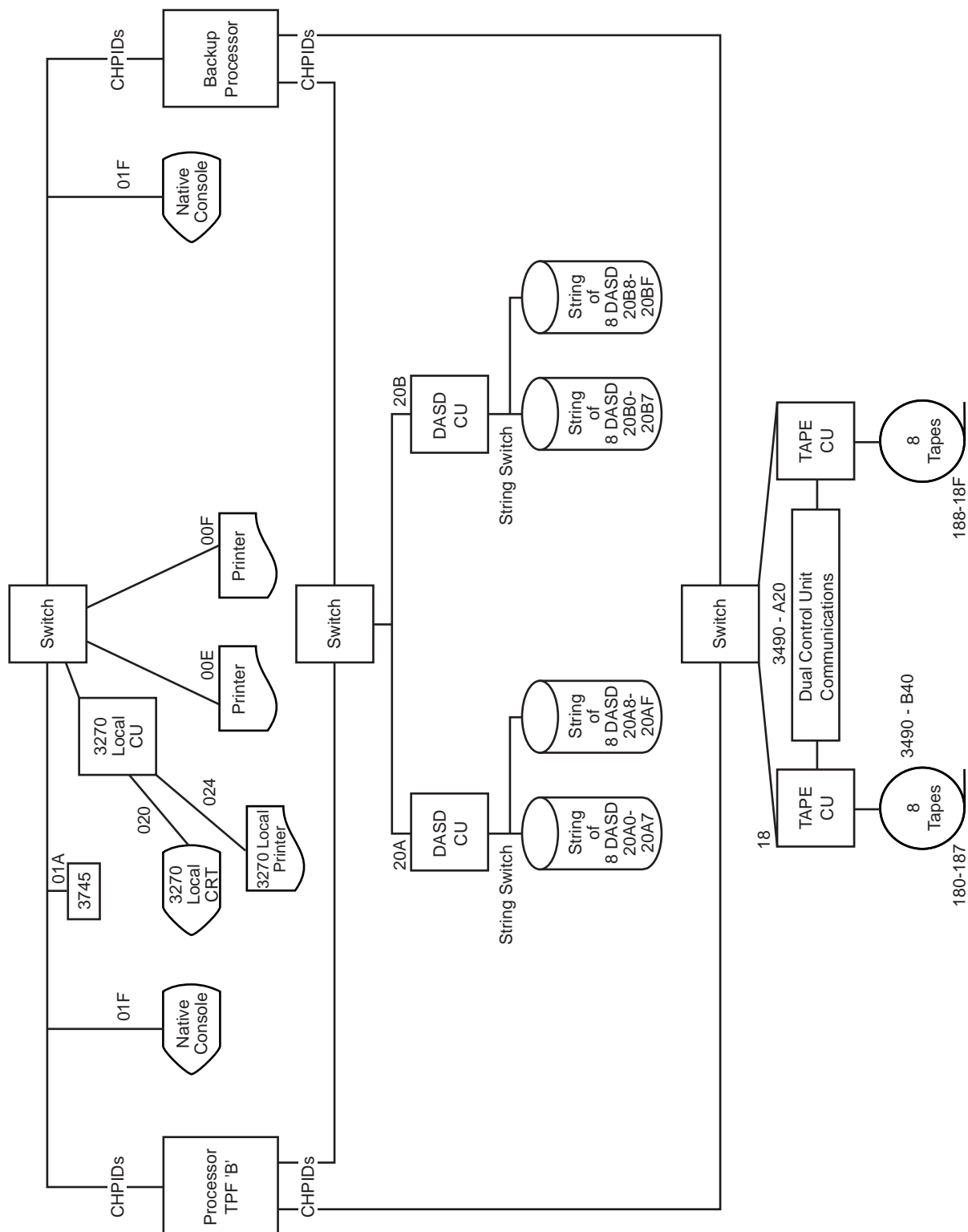


Figure 30. Sample Base-Only System Configuration

---

## Base System Configuration

Figure 30 on page 486 is specified to SIP by the following macros and keywords:

```
CONFIG  APRNT=00E,DCUSV=16,SYSID=B
```

- The system printer is defined as '00E', but it could have been '00F'.
- The type of DASD CU and devices shown utilize up to 16 device addresses.
- The TPF symbolic processor ID is chosen and defaults are accepted for all other keywords.

```
CRASTB  PRCRS=(1F,20),ROCRS=24,NCONSL=YES
```

- This macro defines the address of the system's native console ('01F') that will be used as *prime CRAS* and defines a 3270 local CU that has at least one 3270 local printer ('024') for use as *RO CRAS*.
- Note that on some processors (for example, the 3081) where the native console is not addressable, it is necessary to specify that one of the CRT devices (for example, 020) on the same 3270 local CU as the RO CRAS device is the prime CRAS.
- The user may also optionally select one of the 3270 local CRTs as prime CRAS even if the processor has an addressable native console.
- This same 3270 local CU may contain additional devices (7 CRTs and 7 printers) that can be defined as *alternates* (for example, '020').
- Note that this CU **is not** defined via the LINES and NETWK macros. Those parameters are used to specify **additional** 3270 local control units and terminals.
- The user has the capability to use the 7412 RPQ to attach the 1052/3215 typewriter as a system console to certain processors. Optionally, a 1053-like printer may also be attached as an RO CRAS device or the 1052/3215 may be used for both functions. The user must choose **either** native console or 1052 support at system generation time.

```
UNITRD  UNIT=3505,ADDR=00C,SYMNO=1  
UNITRD  UNIT=3211,ADDR=00F,SYMNO=1  
UNITRD  UNIT=3211,ADDR=00E,SYMNO=2
```

- These macros, one for each U/R device, are specified to include the devices depicted for application program use.
- Note that 00E is optionally included both here and on the CONFIG macro (APRNT) in order to also allow application programs to access the same printer.

IODEV	DVTYP=DASD,IOADR=20A0
IODEV	DVTYP=DASD,IOADR=20A8
IODEV	DVTYP=DASD,IOADR=20B0
IODEV	DVTYP=DASD,IOADR=20B8
IODEV	DVTYP=37X5,IOADR=01A

This series of macros is specified to include the DASD CUs and devices:

- The DVTYP specification can be for any one of the supported DASD device types.
- The logical address is for the first device on the CU as viewed by the active processor.

IODEV	DVTYP=TAPE,IOADR=18
-------	---------------------

This macro specifies 16 tape drives starting with 180 and continuing through 18F. Because there is a maximum of 8 drives per control unit, 2 control units are required: one for drives 180–187 and a second for drives 188–18F. If the communications feature is installed between the two control units and TPF supports multipathing to those devices, then TPF can use both control units to communicate with any of the 16 devices. If, however, the communications feature is not installed between the control units or TPF does not support multipathing for those drives, then only one control unit at a time may be used to communicate with a device.

---

## Loosely Coupled System Configuration

There are a number of major differences between the base-only configuration and the loosely coupled configuration:

- Each processor has its own system console (or its own 3270 local CU with a CRT device) and only certain processor models are supported for inclusion in the complex.
  - The same or different processor models may be included in the complex or as backup processors.
  - If native console support is used each processor **must** have its own 3270 local CU with at least one 328x printer as its RO CRAS device.
  - Either native console or 1052/3215 support may be specified, and all processors must use the same type of devices but they may have different addresses.
  - Processors may be used that have different main storage sizes but the size of the smallest must be specified.
  - Processors may be equipped with a different number of channels but SIP must be coded with the value for the processor with the fewest number.
- If the tape devices are shared, each processor must access the tape devices using the same addresses.
- All of the shared DASD CUs and devices are accessed from each processor via the same addresses. In addition, **all** shared DASD CUs must be equipped with either the LLF or the CFLF RPQ. Depending on the RPQ, the following differences exist:
  - For LLF, all of the shared DASD CUs must have a path to each processor where the interface is installed in the exact same sequence on all CUs (that



is, if 1 DASD CU is connected to the first processor through interface A, **all** other DASD CUs connected to the same processor **must** also use interface A, and interface B would be used to connect to the second processor, and so on).

- For CFLF, unique pathing is not required.
- Each processor has its own unique tape configuration.
- A TCU running 3705/EP is shown to emphasize the significance of an *EP processor*.
- A TCU with multi-tail capability and running ACF/NCP must be connected to all processors and can be on either a byte or block-multiplexer channel but must be located at the same address for all processors.
- Any unit record equipment, in addition to the system printer depicted, must not be shared but may be located at the same addresses on each processor.
- Not shown but required is a Clock Synchronization RPQ that must be connected to each of the processors in the complex.
- Additional processors may be generated but not used in the online system. This is a useful technique when generating for **today** but preparing for **tomorrow**, or for **backup**.
- The requirements for a backup processor are different for a loosely coupled complex and are impacted by cost and the need both for a backup processor and for an EP processor (if there are any 3705/EP protocols being supported).
  - In this illustration the backup processor has been configured in such a way as to be able to replace **either** of the two active processors.
  - Another possible configuration would be to backup each processor in the complex with another processor.
  - A third configuration would be to generate a complex with more processors than are required but to only have active at any one time a subset of the total generated.
- 3270 local CUs, other than one with the system console devices, can only be attached to the EP processor. For configuration purposes, the 3270 local considerations are the same as 3705/EP.
- The input to create a loosely coupled complex is only a single SIP input deck regardless of the number of processors. Thus hardware, which is unique to one processor, must be defined to all but only used by one.

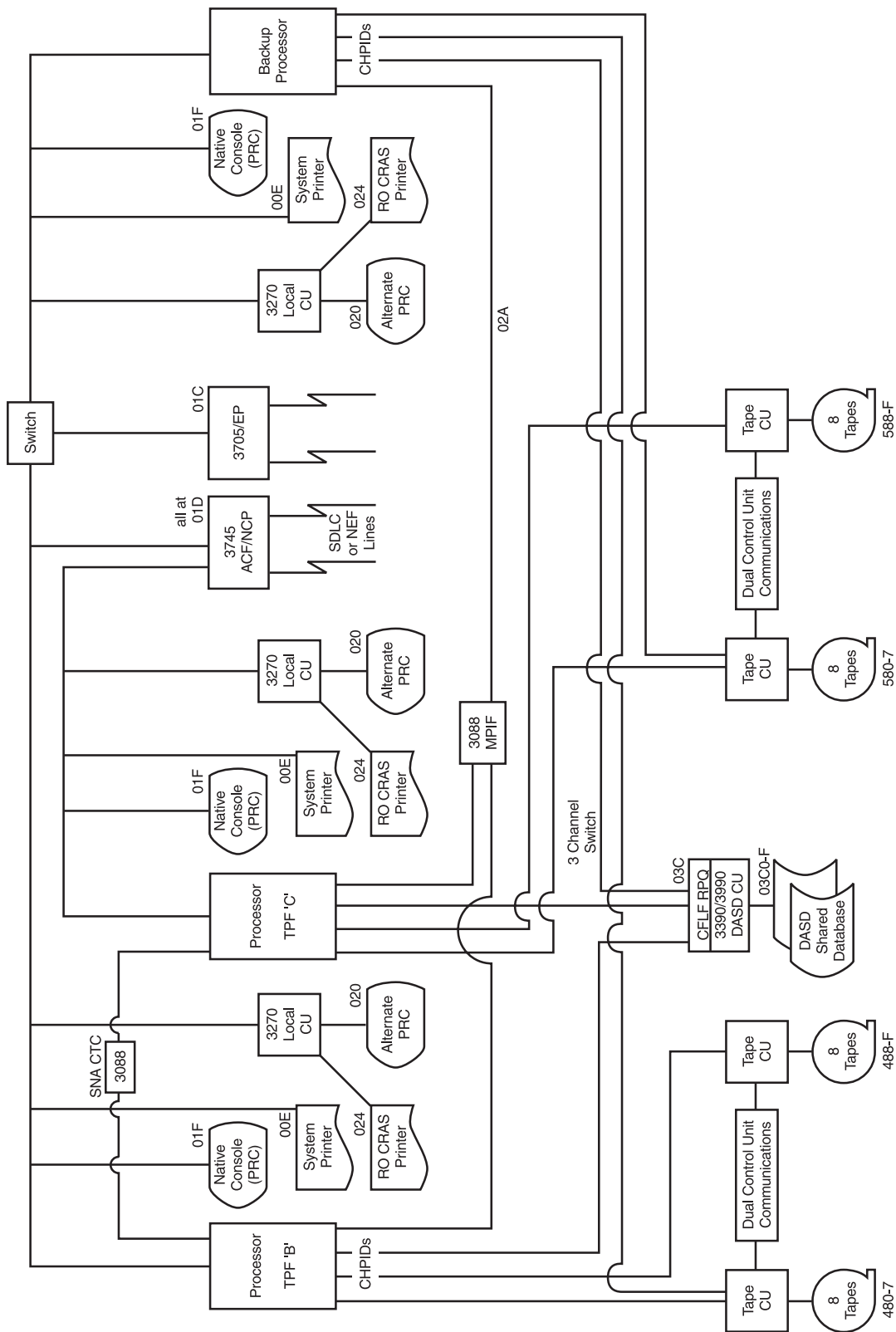


Figure 31. Sample Loosely Coupled System Configuration

The SIP input required for this LC complex is:

CONFIG	APRNT=00E,DCUSV=16,SYSID=(B,C),	X
	PROC1=(xxxxx,yyy,0,0),	X
	PROC2=(wwwwww,zzzz,1,1),	X
	PROC3=(uuuuu,tttt,2,2)	

- The system printer is defined as '00E' and it must be the same for all processors in the complex.
- The type of DASD CU and devices shown utilize up to 16 device addresses. Typically, the number of DASD devices per CU in an LC complex will be less than a base only system due to the performance impact that results from the sharing of CUs (busy conditions).
- The symbolic processor IDs chosen are 'B' and 'C'. This is a critical parameter since it specifies the number of processors in the complex. Additionally, the first subparameter value of SYSID designates the EP processor. Although the illustration implies that SIP assigns the IDs to a particular processor, in reality the choice is selected at restart time. Since they are configured exactly alike, any of the three can assume the identity of either 'B' or 'C'.
- The processor *identification number* must be specified for each of the processors in the complex. The backup processor must be assigned one of these IDs when it is actually utilized.
- The hardware selection address of the Clock Synchronization RPQ must be specified for each processor. The backup processor must be able to access the TOD RPQ using one of the Synch Selection Addresses.
- Defaults are accepted for all other CONFIG keywords.

CRASTB    PRCRS=(1F,1F,20),ROCRS=(24,24),NCONSL=YES
---

Each LC processor must have a system console. In addition, each must have its own 3270 local CU with at least one printer if using native console support.

- Addresses are specified sequentially by processor in the same order that the processors were defined on the CONFIG macro (SYSID).
- All addresses specified are actually available to all processors. All other addresses are treated as alternates by all processors.
- All system consoles must be either 3270-like or 1052/3215 devices. They may also all have alternates. The addresses specified ('01F' and '020') are the same, but they could be different.
- The 3270 local printer ('024') is designated as the RO CRAS device in both cases but are physically different devices. Again, the addresses specified can be different for each processor.

LINES    ... NETWK    N2703=1,SUBCH=(1C-n)
---

- These macros are coded in order to utilize the TCU depicted ('01C').
- The type of protocols and numbers of lines are specified via the LINES macro, whereas the TCU running 3705/EP and is defined via the NETWK macro.

- The addresses, number of TCUs, and symbolic TCU numbers are specified on NETWK along with the channel adapter type.
- Note that the addresses of the TCUs running ACF/NCP are specified on the SNAKEY macro (MAXALS=), **not** here.
- The other keywords on NETWK relate to the protocol being generated and the appropriate ones must be coded or defaults accepted.

```
UNITRD  UNIT=3211,ADDR=00E,SYMNO=1
```

- This macro is specified to include the printer ('00E') for application program use in addition to system use (CONFIG, APRNT).
- If there are any additional U/R devices they could be on any or all of the processors with the same or different addresses. They are included by defining the device to **all** processors. If the same address is used, it is defined only once.

```
IODEV  DVTYP=DASD,IOADR=03C0
IODEV  DVTYP=DASD,IOADR=03C8
IODEV  DVTYP=DASD,IOADR=02A0
```

This series of macros defines the DASD CUs and devices.

- The DVTYP specification can be for any one of the supported DASD device types but must utilize the 3880 CUs with the LLF RPQ or the 3990 CU with the LLF or CFLF RPQ.
- The number of devices on a CU is dependent upon the device type but due to both CU and device contention will probably be less than a base only system configuration.
- All control units defined to the system must have the CFLF or LLF RPQs installed.

```
IODEV  DVTYP=TAPE,IOADR=48
IODEV  DVTYP=TAPE,IOADR=58
```

The tape configuration for each of the processors in the complex is defined as if a single system had multiple tape CUs.

- In this case one macro is for processor 'B' and one is for processor 'C'.
- Both processors can have a tape CU at the same address and in this case only one set of macros is required as if there was only one CU.
- Each macro specifies 16 tape drives. Because there is a maximum of 8 drives per control unit, 2 control units are required per tape subsystem. If the communications feature is installed between the two control units and TPF supports multipathing to those devices, then TPF can use both control units to communicate with any of the 16 devices. If, however, the communications feature is not installed between the control units or TPF does not support multipathing for those drives, then only one control unit at a time may be used to communicate with a device.

---

## Appendix E. SIP Library References

---

### ACP.SYMACRO.RELv v Members

The following members have their source code produced as a result of executing SIP stage II unless otherwise noted.

Macro	Reference
<b>BSSSET</b>	Ensures subsystem environment is consistent with the basic subsystem
<b>BXAXF</b>	File capture data files
<b>C\$GTSZ</b>	GTSZ input header file
<b>C\$IDFUNC</b>	SALO function switch settings
<b>CC0CC</b>	3705 keypoint record
<b>CC1CC</b>	3705 keypoint support
<b>CK9KC</b>	Macro called to expand keypoint record 'B'
<b>CYMZA</b>	CYMZ (MCT) entries for device A (created by FCTBG, copied by SYCON)
<b>CYMZB</b>	CYMZ (MCT) entries for device B (created by FCTBG, copied by SYCON)
<b>CYMZC</b>	CYMZ (MCT) entries for device C (created by FCTBG, copied by SYCON)
<b>CYMZD</b>	CYMZ (MCT) entries for device D (created by FCTBG, copied by SYCON)
<b>FMTA</b>	FMT (formatter) cards for device A (created by FCTBG)
<b>FMTB</b>	FMT (formatter) cards for device B (created by FCTBG)
<b>FMTC</b>	FMT (formatter) cards for device C (created by FCTBG)
<b>FMTD</b>	FMT (formatter) cards for device D (created by FCTBG)
<b>JPC0</b>	Data definitions for Data Reduction Package
<b>LINEQC</b>	Line equate macro
<b>SDAT</b>	Symbolic device address table (for 37x5 control units and SNA CTC connections)
<b>SKGTSZ</b>	Control program CSECT sizes
<b>SSSET</b>	Subsystem dependent global values
<b>STCEQ</b>	Table of record types for STC (System Test Compiler)
<b>SYCON</b>	System configuration definition
<b>SYFCTB</b>	Validation Pass File. Created by FCTBG and used for validation by SIP
<b>SYGLB</b>	Variable symbol definitions
<b>SYGLBK</b>	Variable symbol definitions for communications pilot generation
<b>SYMSEQ</b>	Miscellaneous equate portion of SYSEQC
<b>SYNLST</b>	Global synchronization equate list

<b>SYSEQC</b>	System equates produced by FCTBG
<b>SYSET</b>	Variable symbol values
<b>SYSETK</b>	Variable symbol values for communications pilot generation
<b>UIPROT</b>	Table of utility interface programs

---

## ACP.SYSRCE.RELv<sub>v</sub> Members

The following members have their source code produced as a result of executing SIP stage II.

<b>Member</b>	<b>Reference</b>
<b>CCKP<sub>vv</sub></b>	3705 keypoint record
<b>COHA<sub>,...</sub></b>	RCAT initialization record
<b>COSY<sub>vv</sub></b>	Tape control unit cross reference table
<b>CRS0<sub>vv</sub></b>	BSC station address table
<b>CRS1<sub>vv</sub>, ...</b>	SNCT initialization table
<b>CTKAGF</b>	General file keypoint record 'A'
<b>CTKA<sub>vv</sub></b>	Keypoint record 'A'
<b>CTKC<sub>vv</sub></b>	Keypoint record 'C'
<b>CTKD<sub>vv</sub></b>	Keypoint record 'D'
<b>CTKE<sub>pp</sub></b>	Keypoint record 'E'
<b>CTKI<sub>vv</sub></b>	Keypoint record 'I'
<b>CTKM<sub>vv</sub></b>	Keypoint record 'M'
<b>CTKVGf</b>	General file keypoint record 'V'
<b>CTKV<sub>vv</sub></b>	Keypoint record 'V'
<b>CTKX<sub>vv</sub></b>	Image Pointer Record
<b>CTK0<sub>vv</sub></b>	Keypoint record '0'
<b>CTK6<sub>vv</sub></b>	Keypoint record '6'
<b>CVZD<sub>vv</sub></b>	General file control record
<b>FMTRDECK</b>	Online module initialize and formatter JCL.
<b>IPAT<sub>vv</sub></b>	Assembler source for PAT table.
<b>LOADDECK</b>	System loader - Programmer's Guide
<b>PARS<sub>vv</sub></b>	Release load list
<b>SIGT<sub>vv</sub></b>	Global synchronization table

where

**vv** = Version number where vv is the release version number from the program list (SPPGML).

**pp** = Version number where pp designates the processor for which the keypoint is intended.

## Appendix F. SIP Stage II Data Sets

The following six data sets are required output data sets for SIP stage II. In the chart below, the LRECL, RECFM, and SECONDARY TRACKS should not be changed when allocating these data sets. The values in the other two columns, PRIMARY TRACKS and DIRECTORY BLOCKS, should be changed.

Table 24. Required Output Data Sets for SIP Stage II

Library	LRECL	RECFM	Primary Tracks	Secondary Tracks	Directory Blocks
BDF.V1R1M3.BDFOBJ1.yyyy <sup>5</sup>	80	FB	400	20	40
BDF.V1R1M3.BDFLNK1	80	U	400	20	35
ACP.SYMACRO.RELv <sub>v</sub>	80	FB	200	20	35
ACP.SYSRCE.RELv <sub>v</sub>	80	FB	150	20	35
ACP.SALTBL.RELv <sub>v</sub>	12	FB	20	20	35
TPF.ANTS.RELv <sub>v</sub> <sup>6</sup>	80	FB	20	20	5

### Notes:

1. These numbers are only guidelines.
2. The data sets BDF.V1R1M3.BDFOBJ.yyyy and BDF.V1R1M3.BDFLNK1 are required for users of the TPFDF product.

In a multiple database function (multiple database function) environment, the SIP output data sets will be given an additional level of name qualification. The subsystem name as specified in the SSDEF macro will be appended to the data set name, in this format: ACP.SYMACRO.RELv<sub>v</sub>.ssname.

The following data sets are required for SIP stage II and must be cataloged and allocated as follows:

Library	LRECL	RECFM	Primary Tracks	Secondary Tracks	Directory Blocks
ACP.SRCE.RTPTF.RELv <sub>v</sub>	80	FB	500	20	56
ACP.LINK.OCO	80	U	1000	40	70

This data set will contain all real-time segments updated via a PUT tape. This data set must be allocated and cataloged even if no PUT tape is to be applied before running SIP. The use and maintenance of this data set is described in the Program Directory shipped with the release.

The JCL for allocating TPF data sets is shown in Figure 27 on page 177.

4. vv = Current release version number. Refer to the GENSIP macro OBJLIB parameter.

5. yyyy = The subsystem name.

6. Refer to the GENSIP macro ANTPDS parameter.





---

# Index

## Special Characters

#MAILxx record 107, 110  
#OSIT record 112  
    Open Systems Adapter (OSA)-Express support 112  
#RT1RI record 112  
    HPR support 112  
#RT2RI record 112  
    HPR support 112

## Numerics

3270 local support 123  
3600/4700 SNA generation requirements 140

## A

AAA/RCB initialization table (UAT) 137  
access time minimization 62  
AD/Cycle C/370 requirements 173  
address formats  
    implementation 25  
adjusted I-stream utilization 48, 51  
agent assembly area (AAA) 138  
airlines line control (ALC) support 8  
ALC (airlines line control) support 8  
allocate user programs 261  
allocation of programs 36  
allocator decks 183  
allocator macros 36  
allocator, creating a user 183  
alternate DASD paths 87  
application and system support  
    MDBF considerations 142  
    packages 141  
    TPF collection support 162  
application global area 40  
application program areas 36  
application program interface 141  
APPN support 8  
APSIZ parameter 33  
assembler requirements 173  
automatic cartridge loader  
    filling automatically 55

## B

band number 292  
basic communications support 123  
basic subsystem (BSS) 9  
    definition 9  
BBSAT macro 208  
binary synchronous communication (BSC) 8  
block size design criteria 42  
BSC (binary synchronous communication) 8  
BSC input macros 128  
BSC support 123  
BSNCT macro 210

building a FACE table 187

## C

C language real-time table (CRT) 179  
C language table 179  
C language user real-time table (CUR) 179  
C/370 requirements 173  
C++ language source table (CPP) 180  
cabling for synchronization 227  
capture package 155  
    SIP macros 155  
CCCTIN (control program initializer) 41  
CCPERR macro 212  
CCPPOL macro 214  
CCPSTA macro 216  
CHPID 87  
CIMR component size 330  
clocks 32  
CLOCKS macro 217  
communication control unit (CCU) 132  
communication device loading 26  
communication keypoint records, non-SNA 16  
communication pilot record generation  
    non-SNA communication records 137  
communication support  
    airlines line control (ALC) support 8  
    binary synchronous communication (BSC) 8  
    SNA 8  
    synchronous link control (SLC) 8  
    TCP/IP support 8  
    X.25 support 8  
communications source message router 148  
compiling the FACE Table Generator 188  
computer room agent sets (CRAS) 23  
concurrency filter lock facility (CFLF) 10  
CONFIG macro 219  
configurations, sample 485  
control program  
    area 34  
    fixed record support 7  
    initializer 41  
    pool record support 7  
    records and table area 41  
    table 178  
    user exits 152  
control section (CSECT) 34  
core image restart area component size 330  
core resident 37  
    MDBF allocation considerations 38  
    program area requirements 38  
    program size 38  
    user program allocation 37  
corefast 37  
CORREQ macro 229  
COSY, tape control unit cross-reference table 53, 264, 494  
CP (control program) table 178

- CPP (C++ language source table) 180
- CR allocation 37
- CRAS table macro 130
- CRASTB macro 234
- creating entries in the RIAT, RIATA FINISH 304
- creating entries in the RIAT, RIATA ID 305
- creating entries in the RIAT, RIATA START 311
- CRT (C language real-time table) 179
- CTK0 keypoint 75
- CTK1 keypoint 75
- CTK2 keypoint 75
- CTK3 keypoint 75
- CTK4 keypoint 75
- CTK5 keypoint 75
- CTK6 keypoint 76
- CTK9 keypoint 76
- CTKA communication options 127
  - non-SNA support macros 131
  - SNA requirements 132
- CTKA keypoint 75
- CTKB keypoint 75
- CTKC keypoint 75
- CTKD keypoint 75
- CTKE keypoint 75
- CTKI keypoint 75
- CTKM keypoint 75
- CTKV keypoint 75
- CUR (C language user real-time table) 179
- CX0CK keypoint 330

## D

- DASD
  - alternate paths to 87
- data collection and reduction (DC/DR) 153
- data communications support 123
  - non-SNA 123
  - SNA 123
- data reduction 153
- data set
  - initialization 175
  - SIP stage II 495
- data set initialization 175
- data storage, TPF collection support 162
- database
  - allocation 91
    - application programs, disk resident 91
    - database duplication 98
    - disk file pool storage 92
    - disk resident application programs 91
    - file assignment considerations 92
    - file design 99
    - fixed file data records 91
    - multiple DASD device types 103
  - device addressing 79
    - copy module slots 83
    - file status tables 83
    - restrictions 79
  - device capacity 79
  - loading 22
  - reorganization (DBR) 25, 154

- database (*continued*)
  - input phase 25
  - output phase 25
  - support 57
- DATACO macro 240
- DC/DR
  - DATACO macro 154
- DDCCAP macro 242
- default tape library category 54
- design phase 6
- design, system 5
- device addressing, database 79
- device capacity, database 79
- device independent file addressing
  - implementation 24
  - objectives 24
- DFAD (FACE driver and offline interface) 162
- diagnostic output formatter 159
  - incore dump formatter (ICDF) 159
  - post processor 159
  - terminal simulation 159
- diagrams for macro models xvi
- disk
  - allocation 36
  - file storage 61
  - formatter, real time 161
  - roll call 21
- DLL (dynamic link library) load module table 178
- dynamic link library (DLL) load module table 178

## E

- ECB virtual memory
  - layout of 35
- encryption techniques 140
- entry control block (ECB)
  - work area 141
- execute channel program (EXCP) 78
- executing job stream (stage II)
  - analyzing errors 329
  - macro description 329
- expression enhancements for the TPF debuggers 121
- extended globals
  - support 40
- external lock facility 60

## F

- FACE table generation 186
- fallback pool sections
  - long-term pool sections 71
  - primary fallback 71
  - secondary fallback 71
  - short-term basic pool type 71
- fast recovery table 20
- file address compute program (FACE)
  - driver and offline interface 162
  - program 39
  - table (FCTB) 39
- file address reference words (FARW) 24
- file resident 37

- file resident (*continued*)
  - user program allocation 37
- files
  - auxiliary 156
  - capture and restore 155
  - data 156
  - design 57
  - E-type 156
  - fully duplicated system 62
  - general 155
  - layout 61
  - loaders 155
  - loosely coupled record duplication 60
  - module-to-module duplication 62
  - nonduplicated system 61
  - partial record duplication 62
  - record duplication 60
  - storage, fixed 63
  - storage, pool 68
- fill category
  - setting a minimum level 55
- fixed file
  - distribution technique 58
  - storage 63
- fixed head disk storage allocation 36
- fixed record 290
  - area 59
  - fixed record type base address 65
- FR allocation 37
- functional management message router (FMMR) 143
  - ACF support 143
  - application attributes 143
  - HPO support 144
  - implementation guidelines 144
  - line protocols 143

## G

- general data set
  - command 77
  - volume label 77
- general file
  - GENFIL macro example 79
  - loader online segment residency 39
  - predefined 78
- GENFIL macro 245
- GENSIP macro 247
- getsize program 330
- global
  - application 40
  - area 40
- global area 153
- GLOBAL macro 255
- global macros
  - GLOBAL 40
  - GLSYNC 40
- GLSYNC macro 258
- Greenwich Mean Time (GMT) 162
- GTSZ program 330

## H

- hard IPL 20
- hardware requirements 173
- High Performance Option (HPO) 9
- high-performance routing (HPR) support 112

## I

- I-stream utilization 48, 50
- IBMPAL macro 36, 183
- ICKDSF 175
- ICKDSF disk formatter 175
- ICL (ISO-C link table) 180
- image pointer record (CTKX) 329
- in-core dump formatter (ICDF) 39, 56
- INDSN macro 261
- initialize data sets 175
- initializer for CP 41
- input/output control block (IOB) 41
- installation requirements 173
- IOCDS considerations 87
- IODEV macro 264
- IPL 19
  - program 19
  - with the CLEAR option 21
- ISA (ISO-C assembler function table) 180
- ISC (ISO-C source table) 180
- ISO-C assembler function table (ISA) 180
- ISO-C link table (ICL) 180
- ISO-C source table (ISC) 180

## J

- JCL
  - SIP Stage II 473

## K

- keypoint
  - loading 17
  - macros (non-SNA) 132
  - record 73
  - staging area (KSA) 74
  - status record (PKST) 16, 132
  - table 178
  - X 330
- keypoint backup area (KBA) 74
  - MDBF considerations 74
- keypoint records
  - control program 74
- KP (keypoint) table 178

## L

- life of a message 42
- LIFO devices 54
- limited lock facility (LLF) 10
- line shutdown and restart
  - restart level 127
  - shutdown level 127

- line status table (LSTB) 132
- LINES macro 131, 267
- LISTAPE output 175
- loader general file 18
  - offline segment 19
- log processor 145
  - security levels 145
- LOGCAP macro 269
- logical files 58
- logical unit (LU) identification and control 145
- logical write protect
  - setting 54
- long message transmitter package (LMT) 24, 151
- loosely coupled
  - caution 11
  - description 10

## M

- macro model diagrams xvi
- macros, stage II 330
- main storage
  - boundaries 34
  - global area 153
  - layout 33
    - core definition 33
    - fixed storage (permanent core) 33
    - memory dump 33
    - SIP macros 33
    - storage protection 33
  - resources 31
- mapping support 152
- MAXQ (ZCNIS parameter) 49
- message life 42
- message router support 142
- message switching packages
  - domestic 152
  - World Trade 152
- messages
  - system generation 349
- models of macro invocations xvi
- module file status table (MFST) 21
- MSGRT macro 271
- MSGRTA macro 272
- Multi-Processor Interconnect Facility (MPIF) 11
- multiple database function (MDBF) 9
- multiple TPF images 74
- multiprocessing 12

## N

- native console support (NCS) 123
  - agent assembly area (AAA) 124
  - LNIATA 124
- NETWK macro 279
- non-SNA communications 335
  - keypoint records 16
  - pilot tape creation 345
- nucleus storage boundaries 34

## O

- object code only (OCO) table 180
- OCO (object code only) table 180
- offline
  - interface (DFAD) 162
- offline (OL) table 179
- offline FACE table generation 186
- OL (offline) table 179
- ONLFIL macro 282
- online segment 19
- Open Systems Adapter (OSA)-Express support 112
- operating system requirements 173
- operator control 146
- operator identification and control 145
- optional software support 7
- ordinal numbers 63
- organization 63
- output types 330

## P

- page zero 34
- PAL 36
- paths to DASD, alternate 87
- pilot tapes 24, 137
- PL/I requirements 173
- pool directory
  - generation and maintenance 157
  - minimum 73
- pool file storage
  - bit indicator, pool directory 68
  - duplication 68
  - longevity 68
  - minimum, pool directory 73
  - pool types 69
  - RAMFIL parameters 72
  - record size 68
- pool maintenance program 73
- pool records 290
- precycle above 1052 32
- primary nucleus area 34
- processor
  - definition 31
  - resource ownership table (PROT) 10
  - support 31
  - unique hardware/software 10
- program
  - nesting 42
  - program generation 15
  - program loading 17
    - diagram 18
    - system allocator 17
    - system loader 17
  - tables 36
  - test vehicle (PTV) 157
    - PTV keyword 157
    - support requirements 157
- program tables, modifying 181
- program tables, SPPGML 178
- PSW storage area 34

PU2.1 support 8  
PU5 support 8

## Q

queueing time 57

## R

railroad tracks xvi  
RAM macro 286  
RAMEND macro 289  
RAMFIL macro 290  
RAMFIL statements 65  
RAMFIL USER parameter examples 67  
REACT (ZCNIS parameter) 49  
real-time tables 179  
realtime  
    disk formatter (FMTR) 161  
    substitution 6  
record hold table 41  
record ID attribute table (RIAT) 39  
    description 184  
record size support 57  
    record types 58  
record specification 290  
record types, system utilized 103  
record uniqueness 67  
recoup 154  
requirements, software 173  
RESCAP macro 302  
resource addressing  
    generation 10  
resource vector table (RVT) 140  
restart  
    and switchover 3  
    area  
        core-image records 88  
        size 90  
restore package 155  
    SIP macros 155  
RIATA FINISH macro 304  
RIATA ID macro 305  
RIATA macro 185  
RIATA START macro 311  
RT (real-time) table 179  
RTS (real-time special) table 179

## S

scheduler  
    algorithm 50  
    tightly coupled 48  
SCHID 87  
SCK/PKST communications network specification 337  
SDA 87  
SDLC support 123  
SDMF (standard data message file) 121  
seek time 57  
SIGT (system interprocessing global table) 39  
size and performance 38

size of CIMR components 330  
SLC (synchronous link control) 8  
SLC macro inputs 129  
SLC support 123  
SNA communication 8  
SNA communications keypoint  
    keypoint record 2 (CTK2) 75  
SNA table generation 139  
    OSTG program 139  
    SNAKEY macro 139  
soft IPL 20  
software clocks 31  
software requirements 173  
spare records 290  
specify allocator input data set name and  
    members 261  
specify capture and restore variables 302  
specify GENSIP options for SIP Stage II 247  
specify installation configuration (hardware and  
    software) 219  
specify main storage requirements 229  
specify resource allocation characteristics 286  
specify shutdown and restart levels for non-SNA  
    lines 214  
specify type and format of general file data set 245  
specifying records 290  
SPPBLD macro definitions 181  
SPPGML macro 36  
SPPGML macro definitions 181  
SPPGML program tables 178  
SPRIAT macro 185  
SSA  
    See synchronization selection address  
SSDEF macro 312  
stage I, sample 351  
stage II  
    data sets 495  
    job stream 329  
    job summary, SIP 473  
    macros 330  
    requirements 173  
standard data message file (SDMF) 121  
startup programs, definition 17  
storage  
    boundaries 34  
    control blocks 46  
    director 87  
    initial allocation 43  
        SIP macro parameters 44  
    MPIF 45  
    per message 42  
STR (Sysplex Timer) 162  
STR cabling 227  
subchannel information block 87  
subsystem support services (SSS)  
    encryption techniques 140  
SYCON variables 127  
symbolic device address 87  
symbolic line status table (SLST) 132  
symbolic names 6  
synchronization cabling 227

- synchronization selection address (SSA) 31
- synchronous link control (SLC) 8
- SYNCLK macro 314
- syntax diagrams xvi
- Sysplex Timer (STR) 162
- SYSPUNCH output 175
- system
  - 1052 state 23
  - allocation list 183
  - clocks 162
  - communication keypoints (SCK) 132
    - communication macros 133
    - generation 335, 336
    - macro coding 338, 342, 343
    - SCK/PKST generation package 138
  - definition 3, 31
  - design 5
  - generation error messages 349
  - generation process 15
  - identification 31
  - initialization package (SIP) 3, 15, 16, 167
    - clock macros 32
    - communication functions 16
    - communication macros 126
    - hardware requirements 172
    - macro coding 207
    - report, sample 351
    - SIP macro coding 207
    - software 172
    - Stage I 175
  - interprocessing global table 39
  - IPL program 21
  - message processor (SMP) 148
    - exit routines 148
  - modification facilities 26
  - network architecture (SNA)
    - table generation 139
  - record types 68
  - restart 20
    - initializer program 21
  - restart schedule 21
  - states
    - computer console state 23
    - CRAS state 23
    - message switching state 23
    - normal state 23
    - utility state 23
  - support 141
  - test compiler (STC) 16, 158
    - DRIL file updates 158
    - SDMF file updates 158
  - utilized record types 103
    - fixed file 105
    - miscellaneous 110
    - record initialization categories 104
  - virtual memory
    - layout of 35
  - work blocks 42
- System Generation Services
  - non-SNA macros 335
  - SIP macros 207

- system virtual memory
  - layout of 35
- system work blocks 230

## T

- tape
  - buffer 54
  - configuration 52
  - control unit cross reference table (COSY) 53, 264, 494
  - devices 53
  - drives 52
  - planning 52
  - requirements 53
- tape device
  - filling automatically 55
- tape library category
  - default 54
- tape operations
  - setting logical write protect 54
- TCP/IP communications keypoint
  - keypoint record 2 (CTK2) 75
- TCP/IP support 8
- terminal
  - control 146
  - identification and control 145
  - interchange status table (TITB) 132
- tightly coupled
  - processor support (TC) 8
  - running under VM 50
  - scheduler
    - algorithm 50
    - purpose of 48
    - tuning 48
- TOD (time of day clock) 162
- TOD synch RPQ cabling 227
- TPF 4.1 features
  - High Performance Option (HPO) 9
  - Multi-Processor Interconnect Facility (MPIF) 11
- TPF capabilities 7
- TPF collection support
  - fixed file records 162
  - storing data on a TPF database 162
- TPF design process
  - description 5
  - diagram 6
- TPF Internet mail server support 107, 110
- transfer time 57
- transfer vectors 36
- transmission control unit (TCU) 134
- transmitter package 151

## U

- UFT/FTI concept 66
- UFT/FTI examples 93
- UFTEND macro 317
- UFTFTI statement 318
- UFTI 66
- UI/reservation package (PARS) 149

- UI/reservation package (PARS) *(continued)*
  - enter tables 149
  - entry tables 149
- uniqueness 67
- unit record status table (URST) 56
- unit record support
  - control program link map table 56
  - equipment 55
  - LC considerations 56
  - UNITRD macro 56
- UNITRD macro 321
- unpacking tapes 171
- unsolicited message processor (UMP)
  - broadcast messages 151
  - CODR directories 150
  - routing variations 150
- UR (user real-time) table 179
- USER 66
  - parameter examples 67
- user program allocation 261
- USEREQ macro 322
- USRTPF macro 183
- utilization 48
- UTPROT macro 326

## V

- virtual file access (VFA) 24
- virtual storage
  - layout of 35
- VM
  - running tightly coupled under 50
- volume serial numbers
  - how to code 282, 283
  - TPF VSN example 85
  - VSN migration tool 86, 249

## W

- working storage 41
  - block size design criteria 42
  - ECB life 41
  - five areas 41
  - initial allocation 43
  - MDBF considerations 42
  - message life 42
  - nesting 42
  - per message 42
  - system work blocks 42
- working storage estimating 42
- write protect
  - logical setting 54
- WTC support 183

## X

- X.25 support 8
- XLAD table 138
- XLMA (XMLT assembly area) 138

## Z

- ZRIPL command 20
- ZSTAT U 49
- ZTPLF FILL command 55









File Number: S370/30XX-34  
Program Number: 5748-T14



Printed in the United States of America  
on recycled paper containing 10%  
recovered post-consumer fiber.

SH31-0171-15

