TPF Database Facility

Installation and Customization

Release 1

GH31-0178-09

TPF Database Facility

Installation and Customization

Release 1

GH31-0178-09

Note!

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

Tenth Edition (October 2001)

This is a major revision of, and obsoletes, GH31-0178-08.

This edition applies to Version 1 Release 1 Modification Level 3 of IBM Transaction Processing Facility Database Facility, program number 5706-196, and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

IBM welcomes your comments. Address your comments to:

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

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

Contents

Figures

Tables

Notices

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

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

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

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

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

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

Programming Interface Information

This book is intended to help you customize and tune the IBM TPF Database Facility (TPFDF) product. This book documents information that is Diagnosis, Modification, or Tuning information provided by the TPFDF product.

Attention: Do not use this Diagnosis, Modification, or Tuning information as a programming interface.

Trademarks

The following term is a trademark of the IBM Corporation in the United States or other countries or both:

IBM
OS/390

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

About This Book

This book explains the steps that are necessary to install and customize the TPF Database Facility (TPFDF) product on a Transaction Processing Facility (TPF) or Airline Control System (ALCS) operating environment.

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, structured programming macro (SPM). 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 *Messages (System Error, Online, Offline) and Master Glossary*.

Before You Begin

If you have not already unloaded the files that are necessary to run the TPFDF product, see the *IBM TPF Database Facility Program Directory* for instructions on how to do so.

Who Should Read This Book

This book is for anyone who is responsible for installing and customizing the TPFDF product on a TPF system or ALCS environment.

Conventions Used in the TPFDF Library

The TPFDF library uses the following conventions:

Typography	Examples of Usage
italic	Used for important words and phrases. For example: A *database* is a collection of data. Used to represent variable information. For example: Enter **ZUDFC DISPLAY ID-***fileid*, where *fileid* is the file identifier (ID) of the file for which you want statistics.
bold	Used to represent keywords. For example: Enter **ZUDFC HELP** to obtain help information for the ZUDFC command.

`monospaced`	Used for messages and information that displays on a screen. For example:
	`PROCESSING COMPLETED`
	Used for C language functions. For example:
	`dfcls`
	Used for examples. For example:
	`ZUDFC DISPLAY ID-J5`

bold italic	Used for emphasis. For example: You ***must*** type this command exactly as shown.
CAPital LETters	Used to indicate valid abbreviations for keywords. For example: KEYWord=*option*

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 TPF Database Facility (TPFDF) Books

- *TPFDF Database Administration*, SH31-0175
- *TPFDF Programming Concepts and Reference*, SH31-0179
- *TPFDF Utilities*, SH31-0185.

IBM Transaction Processing Facility (TPF) 4.1 Books

- *TPF Application Programming*, SH31-0132
- *TPF C/C++ Language Support User's Guide*, SH31-0121
- *TPF General Macros*, SH31-0152
- *TPF Operations*, SH31-0162
- *TPF System Generation*, SH31-0171
- *TPF System Installation Support Reference*, SH31-0149.

IBM Airline Control System (ALCS) Books

- *ALCS Installation and Customization*, SH19-6954.

Online Information

- *Messages (System Error, Online, Offline) and Master Glossary*.

How to Send Your Comments

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

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

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

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

  IBM Corporation
  TPF Systems Information Development
  Mail Station P923
  2455 South Road
  Poughkeepsie, NY 12601-5400
  USA
- If you prefer to send your comments by FAX, use this number:
  - United States and Canada: 1 + 845 + 432 + 9788

- Other countries: (international code) + 845 + 432 +9788

Installation Instructions for TPF Users of the TPFDF Product

This chapter discusses the following:
- Customizing the TPFDF product for the TPF system
- Modifying TPF for the TPFDF product
- Using TPFDF user exits
- Preparing TPF programs and installing the TPFDF product
- Loading the TPFDF product on the TPF system
- Activating the TPFDF product.

Customizing the TPFDF Product for the TPF System

Some TPFDF macros require adjustment to suit individual installations. The following sections outline the changes required for TPFDF macros, programs, or utilities, including:
- Setting symbols in the DBLCL macro
- Setting symbols in the ACPDBE macro
- Setting parameters in the C$ACPDBE header file
- Setting parameters in the C$CRUUSR header file.

Setting Symbols in the DBLCL Macro

The DBLCL macro contains the SETB and SETC assembler instructions, which define installation options for the TPFDF product. Table 1 shows how to set the symbols for a TPF operating system. The TPFDF product must be reassembled to reflect any changes to the options.

Note: The DBLCL macro replaces the ACPGBL macro. The ACPGBL macro is included in this shipment to support installations using DSECTs from previous versions of this product. Ensure that the global symbols in the ACPGBL macro for:
- &ACPDBCB
- &ACPDB4K

are set to the same value as in the DBLCL macro.

Use the recommended settings specified in Table 1. Using settings other than those specified may produce unpredictable results.

Table 1. Setting Symbols in the DBLCL Macro for the TPF System

DBLCL Symbol	Description and Settings
&ACPDBCB	Specify block status in GETCC. TPF provides an option to clear all blocks to X'00'. The symbol &SBCLEAR in SYSET controls this action. If &SBCLEAR is set to clear all blocks, &ACPDBCB should be set to **0**; if not, &ACPDBCB must be set to **1**. **0** TPFDF assumes that all blocks obtained with a GETCC macro are cleared to X'00' using &SBCLEAR. **1** TPFDF has to clear the blocks.
&DFMDBF	Indicates if TPFDF is enabled for a multiple database function (MDBF) environment. **0** Non-MDBF **1** MDBF

Table 1. Setting Symbols in the DBLCL Macro for the TPF System (continued)

DBLCL Symbol	Description and Settings
&DF31BIT	Indicates if all customer-written TPFDF application programs are allocated in 31-bit addressing mode. **0** One or more customer-written TPFDF application programs are allocated in 24-bit addressing mode. **1** All customer-written TPFDF application programs are allocated in 31-bit addressing mode. **Note:** If you set &DF31BIT to 1, you must change all TPFDF fast link programs and programs that contain DBDEF macros to 31-bit addressing mode.
&ACPDB4K	Specify block size (pool files). **0** 4-KB pool blocks. **1** No 4-KB pool block support. Set this to 0.
&ACPDBSY	Specify operating system. **0** TPF
&ACPDBID	Used to specify the ID range formula. **0** Use the ID range of #TPFDBID to X'FFFF' (where the value of #TPFDBID is defined in ACPDBE). **1** Use the ID range formula in UWA2. Set this to 0. If you specify 1, you must also write an ID range formula in UWA2 because no ID range formula is provided as part of UWA2 in the TPFDF product.
&ACPDBST	Support for trace facility (supplied by a third party). This symbol defines whether an ENTRC to UFDA is required for ZUDFM OAS entries. **0** Trace interface support required. The user exit UFDA must be written to provide the address of the DBIFB in the *traced* ECB. **1** Trace interface support is not required, the message `UNABLE TO PROCESS` is given in response to ZUDFM OAS entries.
&ACPDBOF	**0** MVS offline system.
&ACPDBFS	Enables TPFDF to create a block trailer and to stamp each block with the time, the command, and so on when a block is filed on DASD. You can display the stamped information using the ZUDFM OAF and ZUFDM OA*/D/HDR commands. **0** Do not create a block trailer. **1** Create a block trailer and stamp each block.
&ACPDBAA	Specify whether ECBs processing ZUDFM commands will use an Agent Assembly Area (AAA). **0** AAA attached on level 1 or RCB attached on level 3. **1** No AAA attached on level 1 or RCB attached on level 3.
&ACPDBRF	The initial setting is 1 (do not change). RLCHA is used to release files.
&ACPDBIN	Determines if a `TRAPC START` should be generated at the beginning of each macro expansion, and `TRAPC COMPLETE` at the end. **Note:** You must provide the `TRAPC` macro. **0** Do not generate trace intercepts. **1** Generate trace intercepts.
&TPFDBMP	Indicates if TPFDF is enabled for multiprocessing in a loosely-coupled environment. **0** Non-multiprocessing **1** Multiprocessing

Table 1. Setting Symbols in the DBLCL Macro for the TPF System (continued)

DBLCL Symbol	Description and Settings
&DB0138	Specifies when message DB0138 is issued to inform you that an application program attempted to access a pool file address that was previously released by another application program. The initial setting is 0. **0** DB0138 message is always issued. **1** DB0138 message is only issued if the file is in hold status.
&DB013E	Specifies whether system error DB013E returns to the application program or exits the entry control block (ECB). The initial setting is 0. **0** After issuing the DB013E system error, the entry control block(ECB) is exited. **1** After issuing the DB013E system error, control returns to the application program.
&DB013A	Specifies if a DB013A system error is issued to inform you that a file was read backward without having full backward chaining defined. The initial and recommended setting is 0. **0** DB013A system error is issued when a file is read backward without full backward chaining defined. **1** DB013A system error is not issued when a file is read backward without full backward chaining defined. **Notes:** 1. Setting symbol &DB013A to 1 may impact overall system performance. If backward chains are not available, TPFDF may need to search forward from the prime block to locate the requested record. This will require a large number of I/Os if there are a large number of overflow blocks. 2. Files that use add current processing (&SW00OP1 bit 2 set to 1) with no chains (&SW00NOC equal to 0) can always be read backwards even if symbol &DB013A is set to 0 and full backward chaining is not defined. See *TPFDF Database Administration* for information about backward chaining. See *TPFDF Programming Concepts and Reference* for more information about reading backward.

&KMNVAL	Specifies the severity value for the following key organization MNOTEs:
	`KEYn ORG NOT ALLOWED IF GENERAL ORG GIVEN`
	`KEYn REQUIRES ORG IF NO GENERAL ORG GIVEN`
	`KEYn HAS ORG, BUT NO KEY1 ORG OR GENERAL ORG`
	The initial setting is 8. You can set a lower value but this may cause MNOTE errors to be overlooked when you assemble TPFDF programs.

&RECERR	Specifies if broken chains should be reported to the printer during recoup phase 1. **0** Print the message. **1** Do not print the message.
&DFREP	Specifies if the TPFDF recoup activity report message (RECP9011I) should be displayed during TPFDF recoup. **0** Does not suppress the message. **1** Suppresses the message.
&DFPAGE	**0** If the ZUDFC DISPLAY command displays more than one page of output, use the CLEAR key to display additional pages. **1** If the ZUDFC DISPLAY command displays more than one page of output, use the ZPAGE command to display additional pages. See *TPF Operations* for more information.

Table 1. Setting Symbols in the DBLCL Macro for the TPF System (continued)

DBLCL Symbol	Description and Settings
&DFRECAB	Specifies if TPFDF recoup should stop or continue if either system error 141302 (file ID found that does not have a DBDEF macro) or 141303 (file ID found that has no recoup information coded in the DBDEF macro): **0** Stop recoup if either system error 141302 or 141303 is issued. **1** Continue recoup if either system error 141302 or 141303 is issued.
&TPFDBDV	A value that is compared to the TPF application timeout counter, which is defined in field PFXATMR in data macro DCTPFX. When processing in detac mode, if the TPF application timeout counter becomes less than or equal to this value, the TPFDF product will issue a TPF DLAYC macro. The TPF DLAYC macro causes the TPF application timeout counter to be reset. You can set the value of &TPFDBDV to any decimal number from 0 to less than the initial setting of the TPF application timeout counter. For example, assume the initial setting for PFXATMR is 50 (that is, fifty 10ms increments for a 500ms total) and &TPFDBDV is set to 10 in the DBLCL macro. The TPFDF product will issue a TPF DLAYC macro during detac processing once PFXATMR has been decremented to 10 or less (or 100ms or less). The initial setting of &TPFDBDV is 0, which bypasses the comparison and prevents the DLAYC macro from being issued. **Note:** Set this symbol to a nonzero value ***only*** if you are receiving TPF system error 000010 while processing in detac mode. Use this symbol with caution because it can affect ECB and system performance.
&TPFOWN	The initial setting is IBM (do not change).
&MISTYPE	An array that allows you to declare as many as 20 record-type prefixes for miscellaneous record-type files. This array is used to identify miscellaneous record-type files for additional validation of the EO# value in a DBDEF macro statement and the &SW00EO# value in a DSECT macro statement. The initial setting is #MISC. Other miscellaneous record-type files can be added to the array as needed.
&INHIDEF	Specifies the default value for the PACKINHI parameter in the DBDEF macro. **COND** The PACKINHI parameter defaults to **COND**. Setting &INHIDEF to COND provides the highest level of protection from recoup missing index LRECs. However, this can increase pool usage and LREC access time during recoup phase 1. **NO** The PACKINHI parameter defaults to **NO**. Setting &INHIDEF to NO provides more flexibility, allowing customers to decide exactly which files will have packing inhibited. However, there is a risk that recoup can miss index LRECs in files that use the PACKINHI=NO default and which also have a forward index path and reuse old pool blocks during a pack operation. See *TPFDF Database Administration* for more information about the PACKINHI parameter and DBDEF macro.
&NOTMC	Specifies whether the TPF Transaction Manager (TM) functions are enabled. The initial setting is 0. **0** The TM functions are enabled. **1** The TM functions are not enabled.

Setting Symbols in the ACPDBE Macro

The ACPDBE macro contains assembler instructions (EQU) that define symbols. Some of the symbols are installation options for the TPFDF product. Many of the equates used in ACPDBE have synonyms (for example, #ACPDBTS is a synonym for #TPFDBTS). Ensure that the same value is coded for both symbols. Table 2 lists the recommended settings for these symbols.

Table 2. Setting Symbols in the ACPDBE Macro

Symbol	**Description and settings**
#TPFDB01 to #TPFDBFF	These define the TPFDF addressing algorithms. Do not change.
#TPFDBID	Specifies the lowest record ID that can be used for TPFDF files. For example, if #TPFDBID is set to X'8000', record IDs X'8000'–X'FFFF' can be used. The initial setting is X'0000' for TPF systems. **Notes:** 1. The lower the value of #TPFDBID, the higher the amount of memory required for TPFDF tables. 2. Although this symbol allows the use of any record ID for a TPFDF file, other restrictions may still apply. For example, in the TPF system, the following IDs are reserved for IBM: • X'0000'–X'00FF' • X'FC00'–X'FFFF'
#TPFDBTS	Specify the DBDEF index table size. The initial setting is ((X'10000' − #TPFDBID) × 4) + 460 + 4. Do not change this value.
#TPFNODE	The number of B^+Tree nodes buffered in memory. For best performance, set this value equal to (2 * *B^+Tree_level_max*) + 1. **Notes:** 1. Where *B^+Tree_level_max* equals the number of levels in the deepest B^+Tree index structure. 2. The initial setting is 10; the minimum is 4. 3. Too high a value can deplete memory, while too low a value can cause thrashing during updates to the B^+Tree index.
#TPFDBFL	Size (in bytes) of the fast-link table. The initial setting is 4096 (do not change).
#TPFDBTB	Maximum number of blocks to be detached by a DBTRD macro or `dftrd` function. The initial setting is 10.
#TPFDBSB	Maximum number of blocks to be detached by a DBSRT macro or `dfsrt` function in sort batch. The initial setting is 10.
#TPFDBOC	GF OS control block. The initial setting is 0 (do not change).
#TPFDBAC	GF ACP control block. The initial setting is 1 (do not change).
#TPFDBMO	Specifies the maximum number of files that an application program can have open at the same time. A value between 5 and 25 will suit most systems. The initial setting is 25.
#TPFDBSW	The size of the SW00SR block. The initial setting is L4 (4095 bytes), the smallest SW00SR block allowed. A 4095 byte block allows as many as five files to be opened without TPFDF having to get another block.
#TPFDBGD	General data set support tag names (do not change).
#DBDCNBR	Maximum IDs for data collection. The initial setting is 1300.
#DBCOL1	Size of the record ID index for data collection. The initial setting is ((X'10000' − #TPFDBID) × 4). Do not change this value.
#DBCOL2	Size of the data area for data collection. The initial setting is (#DBDCNBR × 236). Do not change this value.

Table 2. Setting Symbols in the ACPDBE Macro (continued)

Symbol	Description and settings
#DBCOLL	Used to calculate the space required for data collection. The initial setting is the sum of #DBCOL1 and #DBCOL2, rounded up to a 4-KB boundary; that is, (((#DBCOL1 + #DBCOL2 + 4095) ÷ 4096) × 4096). Do not change this value.
#DBDET_MAX	Maximum number of detached core blocks for all open files in the range 20–256 for an ECB. Blocks exceeding this number use short-term pool files. This number can be increased or decreased depending on system resources. The initial setting is 200.
#TO_RELEASE	Percentage of unmodified detached core blocks in the range 1–100 that are released by the system maintenance routine before a pending buffered block in detac mode is moved to a short-term pool file. The default is 10%. The number of unmodified detached blocks is specified by the #DBDET_MAX set symbol.
#TPFDBG2	The initial setting is TPF (do not change).
#TPFDBG3	The initial setting is V24 (do not change).
#DB2LUFB	The initial setting is OA (do not change).
#DB3LUFB	The initial setting is OAEA (do not change).
#DB4LUFB	The initial setting is OAEX (do not change).
#DF_MAX_DSP	Maximum number of lines that are displayed by the DBDSP macro or `dfdsp` function. The initial setting is 0, indicating that there is no limit.
#DBENUFB	Set to the character on your keyboard that is equivalent to X'4F'.
#DFUTI01	Creating ECBs in TPFDF recoup and the TPFDF capture/restore utility, information and statistics environment (CRUISE) recalculates the distribution of ECBs over the different levels for every so many ordinals as specified. The #DFUTI01 equate determines how often this is done. The initial setting is 500.
#DFUTI02	Creating ECBs in TPFDF recoup and TPFDF CRUISE recalculates the distribution of ECBs over the different levels depending on the ratio of overflows to prime blocks. The #DFUTI02 equate defines this ratio. The initial setting is 50.
#TPFDBD0 to #TPFDBDF	Specifies the data levels to which open files are attached internally. The TPFDF product is shipped to use 15 data levels by default with equates that are sequential from #TPFDBD1 to #TPFDBDF, and then wrapping to #TPFDBD0. **Note:** There is no reason to change the default values. Because the application data levels are preserved across TPFDF calls, the equates will only affect internal data level use in the TPFDF product.
#DBUIO80 to #DBUIO85	If your system versions of UIO and UIS do not support scrolling, set these symbols to X'C8'. If your UIO and UIS programs support special scrolling, TPFDF programs use these symbols to set the UI2MG2 field in the UIO parameter block to UI2PF and you can set these fields to X'80'–X'85'.
#DBUIOC8	If your system versions of UIO and UIS do not support scrolling, set this symbol to X'C8'.

Setting Parameters in the C$ACPDBE Header File

The C$ACPDBE header file defines the value for the _TPFDBID symbol.

Table 3. Setting Symbols in the C$ACPDBE Header File

Symbol	Description and settings
_TPFDBID	Specifies the lowest record ID that can be used for TPFDF files. For example, if _TPFDBID is set to X'8000', record IDs X'8000'–X'FFFF' can be used. The initial setting is X'0000' for TPF systems. **Notes:** 1. The lower the value of _TPFDBID, the higher the amount of memory required for TPFDF tables. 2. Although this symbol allows the use of any record ID for a TPFDF file, other restrictions may still apply. For example, in the TPF system, the following IDs are reserved for IBM: • X'0000'–X'00FF' • X'FC00'–X'FFFF'

Setting Parameters in the C$CRUUSR Header File

The C$CRUUSR header file contains user-specific settings that you must set before you compile the capture/restore utility, information and statistics environment (CRUISE). The C$CRUUSR header file also defines parameter table default values. These default values are a primary set of values that you can modify when CRUISE is installed. After you have set these values, see “Creating a Parameter Table Index” on page 15 for more information.

Table 4. Parameter Table Defaults

Parameter	Description and settings
CRU_DEF_ECB	The entry control block (ECB) start value as specified by a percent in the range 1–100. The initial setting is 10%.
CRU_DEF_IMB	Indicates if combinations used with parameters WID and ADR select embedded references. The initial setting is Y. **Y** YES **N** NO
CRU_DEF_NLM	Specifies the number of messages to log. The initial setting is −1. **−1** specifies that all messages are logged. *log* specifies the number of messages to be logged, where *log* is a number in the range 0–100.
CRU_DEF_NPM	Specifies the number of messages to print. The initial setting is −1. **−1** specifies all messages are to be printed. *prt* specifies the number of messages to be printed, where *prt* is a number in the range 0–100.

Table 4. Parameter Table Defaults (continued)

Parameter	Description and settings
CRU_DEF_RST	Specifies the restore option. The initial setting is R. **N** specifies that the captured fixed and pool files are restored to new pool file addresses where the data structure is rebuilt. **O** specifies that the database is restored to the same fixed and pool file addresses that were captured. **R** specifies that the captured fixed files are restored to the same ordinals that were captured. The captured pool files are restored to new pool file addresses and the database structure is rebuilt. The old pool file addresses are not released when the structure is rebuilt. **Z** specifies that the captured fixed files are restored to the same ordinals that were captured. The captured pool files are restored to new pool file addresses and the database structure is rebuilt. The old pool file addresses are released when the data structure is rebuilt. **X** specifies that the captured fixed and pool files are restored to new pool file addresses and the data structure is not rebuilt.
CRU_DEF_STA	Specifies if statistics are generated. The initial setting is N. **Y** YES **N** NO
CRU_DEF_SET	Indicates if all pool addresses found are set in use in the pool directory. The initial setting is N. **Y** YES **N** NO
CRU_DEF_TAP	Specifies the default tape name. The initial setting is BFA.
CRU_DEF_TAR	Specifies the target system for a capture. The initial setting is T. **T** TPF **M** MVS
CRU_LOD_V	Does not apply to the TPF system. The initial setting is 0.
CRU_PFA_X_REF	Indicates if all CRUISE functions create an IRCFDF cross-reference prime file while chain chasing. The initial setting is 0. **1** YES **0** NO

Modifying the TPF System for the TPFDF Product

This section describes the adjustments that you must make to your TPF system when installing the TPFDF product on the TPF system.

Follow the instructions in this section to:
- Define fixed files
- Define pool files
- Enable the TPF system for the TPFDF product.

Note: Multiple database facility (MDBF) users must repeat these instructions for each subsystem.

Defining Fixed Files

Define the new fixed file types listed in Table 7 on page 12 and Table 8 on page 13. Add them to the FACE table by coding RAMFIL statements in a SIP stage 1 deck and running the FACE table generator (FCTBG). See *TPF System Generation* for more information.

Note: For an MDBF system, allocate the fixed files specified as ***unique*** in the figures as subsystem user unique.

Defining Pool Files

Use the RIATA macro in SPRIAT to add RIAT characteristics for the pool files shown in Table 8 on page 13. Define the pool file characteristics as shown in Table 8. Add any additional characteristics desired by your installation.

Add RIAT characteristics for pool files in the range X'FCDC'–X'FCFF' as shown in Table 5, and change the device as necessary to include only device types available on your TPF system.

Note: Reassemble the RIAT table after adding RIATA macros.

Table 5. Pool Files

RECORD ID	SIZE	TYPE	DEVICE
FCDC	381	ST	A
FCDD	381	LT	A
FCDE	381	LD	A
FCDF	381	ST	B
FCE0	381	LT	B
FCE1	381	LD	B
FCE2	381	ST	C
FCE3	381	LT	C
FCE4	381	LD	C
FCE5	381	ST	D
FCE6	381	LT	D
FCE7	381	LD	D
FCE8	1055	ST	A
FCE9	1055	LT	A
FCEA	1055	LD	A
FCEB	1055	ST	B
FCEC	1055	LT	B
FCED	1055	LD	B
FCEE	1055	ST	C
FCEF	1055	LT	C
FCF0	1055	LD	C
FCF1	1055	ST	D
FCF2	1055	LT	D
FCF3	1055	LD	D

Table 5. Pool Files (continued)

RECORD ID	SIZE	TYPE	DEVICE
FCF4	4095	ST	A
FCF5	4095	LT	A
FCF6	4095	LD	A
FCF7	4095	ST	B
FCF8	4095	LT	B
FCF9	4095	LD	B
FCFA	4095	ST	C
FCFB	4095	LT	C
FCFC	4095	LD	C
FCFD	4095	ST	D
FCFE	4095	LT	D
FCFF	4095	LD	D

Using TPFDF User Exits

TPF systems have TPFDF user exits for user-specific processing. See *TPF System Installation Support Reference* for more information.

Preparing TPF Programs and Installing the TPFDF Product

To prepare TPF programs and install the TPFDF product, do the following:
- Update the SIP stage I deck
- Run the FACE Table Generator (FCTBG)
- Assemble the stage I deck
- Run the SIP stage II deck

Note: Multiple database facility (MDBF) users must repeat these instructions for each subsystem.

Update the SIP Stage I Deck

1. To enable the TPF system for the TPFDF product, set the value of the TPFDF parameter in the CONFIG macro to **TPFDF=YES**.
2. Specify the EXPRS parameter of the GENSIP macro with the jobs listed in Table 6.

Table 6. Stage I Deck Job Steps to Install the TPFDF Product

Job	Task
F2	Updates macro and SIP support libraries.
G5	Assembles RIAT.
G6	Updates data reduction.
H	Assembles and link-edits the JCL for the TPFDF product.
I1	Assembles control program CSECTs.
I2	Link-edits the control program load module (CPS0).
I3	Assembles and link-edits control programs.
JA	Compiles C functions for the C load module.

Table 6. Stage I Deck Job Steps to Install the TPFDF Product (continued)

Job	Task
JB	Assembles C functions and C++ load modules.
JC	Assembles C++ load modules.
JD	Compiles C functions for the C load module for the TPFDF product.
JE	Assembles C functions for the C load module for the TPFDF product.
JF	Compiles C++ functions for the C load module for the TPFDF product.
J1A	Compiles the GTSZ segment.
J3A	Generates the CTKX segment.
J4	Assembles real-time programs.
J5	Assembles WTC offline components.
J6	Link-edits WTC offline components.
J7	Assembles of user real-time programs.
J8	Compiles C real-time programs.
J9	Compiles user C real-time programs.
J10	Assembles real-time programs for the TPFDF product.
L2	Updates and assembles release PARS lists for the TPFDF product and the TPF 4.1 system.
L5	Link-edits C load modules.
L6	Link-edits C load modules for the TPFDF product.
S	Runs system allocator and creates IPAT.

See *TPF System Generation* for more information about the EXPRS parameter.

Run the FACE Table Generator (FCTBG)

See *TPF System Generation* for more information about running the FACE table generator (FCTBG).

Assemble the SIP STAGE I Deck

See *TPF System Generation* for more information about assembling the SIP stage I deck.

Run the SIP STAGE II Deck

See *TPF System Generation* for more information about running the SIP stage II deck.

Loading the TPFDF Product on the TPF System

When you load the TPFDF product on the TPF system, load the following:
- Control program
- FACE table
- IPAT
- PARS13
- PARS40
- RIAT

For more information about how to load programs, DLMs, and TPF components on a TPF operating system, see *TPF System Generation*.

Note: Multiple database facility (MDBF) users must repeat these instructions for each subsystem.

Activating the TPFDF Product

The following provides detailed steps to get the TPFDF product into operational status.

Note: Multiple database facility (MDBF) users must repeat these instructions for each subsystem.

Initialize the TPFDF Files

Attention: Do not initialize the files listed in Table 7 or in Table 8 on page 13 if you have already installed the TPFDF product on this TPF system because you will lose the current contents of these files.

If the TPFDF product will be used in a multiple database function (MDBF) environment, all TPFDF files must be initialized in every subsystem and, for those that are subsystem user unique, for every subsystem user.

Table 7. Control Files Required by TPFDF

DSECT	RECORD TYPE	REC ID	SIZE	NBR	ORDINAL	SSU *
GR31SR	#IDFUS	FDFF	L1	1	#GR31SRF-#GR31SRL	unique
GR0VSR	#IDFU4	FDFB	L4	1	#GR0VSRF-#GR0VSRL	unique
GR0YSR	#IDFU4	FDF9	L4	1	#GR0YSRF-#GR0YSRL	unique
GR27SR	#IDFU4	FDF8	L4	1	#GR27SRF-#GR27SRL	unique
Note: * In a Non-MDBF environment define all the records as common.						

Initializing the TPFDF Control Files Using ZAREC

Before you can use TPFDF commands (including ZUDFM OAINIT), you must initialize some control files on the real-time database. These files, shown in Table 7, are initialized using the TPF command, ZAREC.

1. Initialize GR31SR (ordinal number 0, file type #IDFUS, and record ID FDFF).

```
ZAREC LIDFUS.0 0 FDFF0000C3E5C2C3

ZAREC LIDFUS.0 8 0000000000000000

ZAREC LIDFUS.0 10 001A000000000000
```

2. Initialize GR0VSR (ordinal number 0, file type #IDFU4, and record ID FDFB).

```
ZAREC LIDFU4.0 0 FDFB0000C3E5C2C3

ZAREC LIDFU4.0 8 0000000000000000

ZAREC LIDFU4.0 10 001A000000000000
```

3. Initialize GR0YSR (ordinal number 1, file type #IDFU4, and record ID FDF9).

```
ZAREC LIDFU4.1 0 FDF90000C3E5C2C3

ZAREC LIDFU4.1 8 0000000000000000

ZAREC LIDFU4.1 10 001A000000000000
```

4. Initialize GR27SR (ordinal number 2, file type #IDFU4, and record ID FDF8).

```
ZAREC LIDFU4.2 0 FDF80000C3E5C2C3

ZAREC LIDFU4.2 8 0000000000000000

ZAREC LIDFU4.2 10 001A000000000000
```

TPF Operations provides a full description of the ZAREC command.

Initializing Noncontrol Files Using ZUDFM

Use the ZUDFM OAINIT command to initialize the fixed files described in Table 8.

See *TPFDF Utilities* for more information about how to use ZUDFM OAINIT and other ZUDFM commands.

Table 8. Non-Control Files Required by TPFDF

DSECT	RECORD TYPE	REC ID	SIZE	NBR	ORDINAL/TYPE	SSU [2]
GR0ZSR[3]	#GR0ZSR	FDF4	L2	30	0–29 per processor	common
GR25SR	#IDFU4	FDF6	L4	1	#GR25SRF–#GR25SRL	unique
GR26SR	POOL	FD00	L4	N/A	LT	unique
GR28SR	#IDFC4	FDF7	L4	1	#GR28SRF–#GR28SRL	common
GR29SR	POOL	FD01	L4	N/A	LT	common
GR3MSR	#GR3MSR	FDF3	L2	99	0–98	common
GR3NSR	POOL	FD03	L4	N/A	LT	common
GR3OSR	#IDFC4	FDF5	L4	1	#GR3OSRF–#GR3OSRL	common
GR32SR	#IDFCL	FDFD	L2	1	#GR32SRF–#GR32SRL	common
IRA5DF	#IDFC4	FDA5	L4	1	#IRA5DFF–#IRA5DFL	common
IRCADF	#IDFC4	FD1C	L4	1	#IRCADFF–#IRCADFL	common
IRCBDF	#IRCBDF	FD1D	L4	200	One Ordinal per CRUISE ordinal table	common
IRCCDF	#IDFC4	FD1E	L4	1	#IRCCDFF–#IRCCDFL	common
IRCDDF	#IDFC4	FD1F	L4	1	#IRCDDFF–#IRCDDFL	common
IRCEDF	POOL	FD21	L4	N/A	LT	common
IRCFDF [3]	#IRCFDF	FD16	L4	499	0–498 per processor	common
IRCGDF [3]	#IRCGDF	FD17	L4	499	0–498 per processor	common
IRCHDF	#IRCHDF	FD18	L4	32	0–31	common
IRCIDF [3]	#IRCIDF	FD19	L4	499	0–498 per processor	common
IRCJDF	#IRCJDF	FD1A	L4	32	0–31	common
IRCKDF	#IRCKDF	FD1B	L4	200	One Ordinal per CRUISE ordinal table	common
IRCMDF [4]	#IRCMDF	FD22	L4	997	0–996 per processor	common
IRCTDF	POOL	FD23	L4	N/A	LT	common
IRCXDF	POOL	FD24	L4	N/A	LT	common

Table 8. Non-Control Files Required by TPFDF (continued)

DSECT	RECORD TYPE	REC ID	SIZE	NBR	ORDINAL/TYPE	SSU [2]
IRDCDF	POOL	FD0A	L4	N/A	LT	common
IRDIDF [1]	#IRDIDF	FDE5	L4	256	0–255 per (processor+1)	common
IRDKDF	#IDFCS	FDE4	L1	1	#IRDKDFF–#IRDKDFL	common
SRCK1P	POOL	FD10	L2	N/A	LT	common
SRHH1P	#SRHH1P	FDDF	L4	10	0–9	common
SRMP1A	#SRMP1A	FDE0	L4	10	0–9	common
SRM31A	#SRM31A	FDDC	L4	10	0–9	common
SRM41A[1]	#SRM41A	FDDE	L4	59	0–58 per processor up to the first 8 processors	common
SRM51A[1]	#SRM51A	FDDD	L4	59	0–58 per processor up to the first 8 processors	common
SRM61A	#SRM61A	FDE1	L4	10	0–9	common
SR54BA	#IDFC4	FDDB	L4	1	#@SR54BA	common
IR00DF	**POOL**	**FD08**	**L1**	**N/A**	**LT**	**common**
IR01DF	**#IDFCL**	**FDE1**	**L2**	**1**	**#IR01DFF–#IR01DFL**	**common**
IR02DF	**#IR02DF**	**FDE7**	**L1**	**26**	**0–25**	**common**
IR03DF	**#IR03DF**	**FDE6**	**L1**	**26**	**0–25**	**common**
IR04DF	**POOL**	**FD09**	**L1**	**N/A**	**LT**	**common**

Note: IR00DF–IR04DF are used for TPFDF education.

Note: [1] This is a partitioned file and the ordinals are spread out over the partitions. The number of ordinals listed is the number for each partition. Set the number of partitions in the DSECT to 1 plus the number of loosely-coupled processors.

Note: [2] In a non-MDBF environment, define all the records as common.

Note: [3] This is a partitioned file and the ordinals are spread out over the partitions. The number of ordinals listed is the number for each partition. Set the number of partitions in the database definition (DBDEF) macro to the number of loosely coupled processors.

Note: [4] Use the following formula to calculate the number of ordinals when using CRUISE with MRC support:

`#ord = prime (mr / 267 / avgch)`

where,

#ord	is the ordinal, which is a prime number, to use for the #IRCMDF file.
prime	is the function to calculate the closest prime number.
mr	is the maximum estimated number of multiple references in a file. You can use the RCI count from the recoup phase 1 activity log as a reference.
267	specifies the number of LRECs that are in a data block for the MRC subfile.
avgch	is the average number of overflow blocks for each ordinal number in the #IRCMDF database. When using multiple reference check (MRC) support, the recommended number of chains to balance performance and resource (fixed files) requirements is 1.5. The greater the *avgch* value, the fewer ordinal (fixed files) requirements needed and the more CRUISE capture and restore processing time increases. If you decrease the *avgch* value, the ordinal (fixed files) requirements needed increases and CRUISE capture and restore processing time decreases. See your database administrator for more information about specifying the *avgch* value. See *TPFDF Utilities* for more information about MRC.

Initialize Command Help

Enter **ZUDFM OAHINIT** to initialize the help facility, which allows you to get online help for commands.

Initialize Database Information Displays

Enter **ZUDFM OAI/DBTAB/REL/CREATE** to initialize the relations database, which allows you to display information about the relationship of files in the database. After the TPFDF product is initialized, enter the command any time you add or change a file on the TPFDF database.

Creating a Parameter Table Index

To create a parameter table index, do the following:

- Enter the ZFCRU EXPAND command to create a parameter table index.
- Enter the ZFCRU DEFAULT command to set the initial default values. See *TPFDF Utilities* for more information about setting default values on a parameter table.

Defining Tape Labels for the TPFDF Product

The following tapes, used by the TPFDF product, require that tape labels are defined. This is done with the TPF command, ZTLBL. See *TPF Operations* for more information.

SPR	input/output
BFA	input/output

Creating the Macro Label Set (MLS) Program

To run the offline MLS program, see member MLS in the sample JCL that is shipped with the TPFDF product. The sample JCL assembles the data macros and the runs segment UF0F, which creates the MLS.

Note: UF0F is assembled and link-edited by SIP.

Defining the Database Definition (DBDEF) Program

The DBDEF program must be defined in main storage before being accessed by an application. To ensure this, define the DBDEF programs using the PRELOAD option in IBMPAL. Otherwise, DBDEF programs are defined using E-type loaders and could be treated as file resident by the TPFDF product if the core resident program areas are full.

Installation Instructions for ALCS Users of the TPFDF Product

This chapter discusses the following:
- Customizing the TPFDF product for the ALCS environment
- Modifying the ALCS environment for the TPFDF product
- Enabling C language support
- Activating the TPFDF product.

Note: ALCS is also referred to as TPF/MVS.

Customizing the TPFDF Code for Your Installation

Some TPFDF macros and a TPFDF header file require adjustments for individual installations. The following sections outline the changes required for TPFDF macros, programs, utilities, or the header file, including:
- Setting symbols in the DBLCL macro
- Setting symbols in the ACPDBE macro
- Setting parameters in the C$ACPDBE header file
- Setting parameters in the C$CRUUSR header file
- Setting symbols in the BLKSZ macro.

These macros are found in the data set with the trailing qualifer, BDFMAC1. The header file is found in the data set with the trailing qualifier, BDFHDR1. If they need to be changed, use a user modification (SMP/E USERMOD).

Setting Symbols in the DBLCL Macro

The DBLCL macro contains SETB and SETC instructions, which define installation options for the TPFDF product. Table 9 describes the symbols defined in the DBLCL macro and provides information about the recommended settings.

Note: The DBLCL macro replaces the ACPGBL macro. The ACPGBL macro is included in this shipment for those installations that have DSECTs that require it.

Table 9. Setting Symbols in DBLCL for ALCS

DBLCL Symbol	Description and Setting
&ACPDBCB	**0** GETCC clears blocks to zeros.
&DFMDBF	Indicates if TPFDF is enabled for an MDBF environment. **0** Always 0 for ALCS (no MDBF)
&DF31BIT	Indicates if all customer-written TPFDF application programs are allocated in 31-bit addressing mode. **1** Always 1 for ALCS (all customer-written TPFDF application programs are allocated in 31-bit addressing mode).
&ACPDB4K	**0** 4-KB block support (pool files).
&ACPDBSY	Specify online operating environment. **1** ALCS

Table 9. Setting Symbols in DBLCL for ALCS (continued)

DBLCL Symbol	Description and Setting
&ACPDBID	Used to specify the ID range formula. **0** Use the ID range of #TPFDBID to X'FFFF' (where the value of #TPFDBID is defined in ACPDBE). **1** Use the ID range formula in UWA2. Set this to 0. If you specify 1, you must also write an ID range formula in UWA2 because no ID range formula is provided as part of UWA2 in the TPFDF product.
&ACPDBST	**0** Support for the ALCS trace facility.
&ACPDBOF	Offline operating environment. **0** The initial setting is MVS.
&ACPDBFS	Enables TPFDF to create a block trailer and to stamp each block with the time, the command, and so on when a block is filed on DASD. You can display the stamped information using the ZUDFM OAF and ZUFDM OA*/D/HDR commands. **0** Do not create a block trailer. **1** Create a block trailer and stamp each block.
&ACPDBAA	Specify whether ECBs processing ZUDFM commands will use an Agent Assembly Area (AAA). **0** AAA attached on level 1 or RCB attached on level 3. **1** No AAA attached on level 1 or RCB attached on level 3.
&ACPDBRF	**1** Use RLCHA to release files.
&ACPDBIN	Determines if a `TRAPC START` should be generated at the beginning of each macro expansion, and `TRAPC COMPLETE` at the end. **Note:** You must provide the `TRAPC` macro. **0** Do not generate trace intercepts. **1** Generate trace intercepts.
&TPFDBMP	**0** No multiprocessor support required.
&DB0138	Specifies when message DB0138 is issued to inform you that an application program attempted to access a pool file address that was previously released by another application program. The initial setting is 0. **0** DB0138 message is always issued. **1** DB0138 message is only issued if the file is in hold status.
&DB013E	Specifies whether system error DB013E returns to the application program or exits the entry control block (ECB). The initial setting is 0. **0** After issuing the DB013E system error, the entry control block (ECB) is exited. **1** After issuing the DB013E system error, control returns to the application program.

Table 9. Setting Symbols in DBLCL for ALCS (continued)

DBLCL Symbol	Description and Setting
&DB013A	Specifies if a DB013A system error is issued to inform you that a file was read backward without having full backward chaining defined. The initial and recommended setting is 0. **0** DB013A system error is issued when a file is read backward without full backward chaining defined. **1** DB013A system error is not issued when a file is read backward without full backward chaining defined. **Notes:** 1. Setting symbol &DB013A to 1 may impact overall system performance. If backward chains are not available, TPFDF may need to search forward from the prime block to locate the requested record. This will require a large number of I/Os if there are a large number of overflow blocks. 2. Files that use add current processing (&SW00OP1 bit 2 set to 1) with no chains (&SW00NOC equal to 0) can always be read backwards even if symbol &DB013A is set to 0 and full backward chaining is not defined. See *TPFDF Database Administration* for information about backward chaining. See *TPFDF Programming Concepts and Reference* for more information about reading backward.

&KMNVAL	Specifies the severity value for the following key organization MNOTEs:
	`KEYn ORG NOT ALLOWED IF GENERAL ORG GIVEN`
	`KEYn REQUIRES ORG IF NO GENERAL ORG GIVEN`
	`KEYn HAS ORG, BUT NO KEY1 ORG OR GENERAL ORG`
	The initial setting is 8. You can set a lower value but this may cause MNOTE errors to be overlooked when you assemble TPFDF programs.

&RECERR	Does not apply to ALCS systems.
&DFREP	Does not apply to ALCS systems.
&DFPAGE	Does not apply to ALCS systems.
&DFRECAB	Does not apply to ALCS systems.
&TPFDBDV	When processing in detac mode, this value indicates if a LODIC CONDITIONALDEFER macro is issued when a block is added or reomved from the detac list. The LODIC CONDITIONALDEFER macro will defer the ECB if it is approaching an application timeout (system error 000010). **0** The LODIC CONDITIONALDEFER macro will not be issued. **1** The LODIC CONDITIONALDEFER macro will be issued. The initial setting is 0. **Note:** Set this symbol to a nonzero value ***only*** if you are receiving TPF system error 000010 while processing in detac mode. Use this symbol with caution because it can affect ECB and system performance.
&TPFOWN	The initial setting is IBM (do not change).
&MISTYPE	An array that allows you to declare as many as 20 record-type prefixes for miscellaneous record-type files. This array is used to identify miscellaneous record-type files for additional validation of the EO# value in a DBDEF macro statement and the &SW00EO# value in a DSECT macro statement. The initial setting is #MISC. Other miscellaneous record-type files can be added to the array as needed.
&INHIDEF	Does not apply to ALCS systems.
&NOTMC	Specifies whether the TPF Transaction Manager (TM) functions are enabled. The initial setting is 0. **0** The TM functions are enabled. **1** The TM functions are not enabled.

Setting Symbols in the ACPDBE Macro

The ACPDBE macro contains instructions (EQU) that define symbols. Some of the symbols are installation options for the TPFDF product. Many of the equates used in ACPDBE have synonyms (for example, #ACPDBTS is a synonym for #TPFDBTS). Ensure that the same value is coded for both symbols. Table 10 describes the symbols defined in ACPDBE and provides information about the recommended settings.

Table 10. Setting Symbols in the ACPDBE Macro

Symbol	Description and settings
#TPFDB01 to #TPFDBFF	Used to define the TPFDF addressing algorithms (do not change).
#TPFDBID	Specifies the lowest record ID that can be used for TPFDF files. For example, if #TPFDBID is set to X'8000', record IDs X'8000'–X'FFFF' can be used. The initial setting is X'0000' for ALCS environments. **Note:** The lower the value of #TPFDBID, the higher the amount of memory required.
#TPFDBTS	Specifies the DBDEF table size. The initial setting is ((X'10000' − #TPFDBID) × 4) + 460 + 4 (do not change).
#TPFNODE	The number of B^+Tree nodes buffered in memory. For best performance, set this value equal to (2 * *B^+Tree_level_max*) + 1. **Notes:** 1. Where *B^+Tree_level_max* equals the number of levels in the deepest B^+Tree index structure. 2. The default is 10; the minimum is 4. 3. Too high a value can deplete memory, while too low a value can cause thrashing during updates to the B^+Tree index.
#TPFDBFL	Size (in bytes) of the fast-link table. The initial setting is 4096 (do not change).
#TPFDBTB	Maximum number of blocks to be detached by a DBTRD macro or `dftrd` function. The initial setting is 10.
#TPFDBSB	Maximum number of blocks to be detached by a DBSRT macro or `dfsrt` function in sort batch. The initial setting is 10.
#TPFDBOC	GF OS control block. The initial setting is 0 (do not change).
#TPFDBAC	GF ACP control block. The initial setting is 1 (do not change).
#TPFDBMO	Specifies the maximum number of files that an application program can have open at the same time. A value between 5 and 25 will suit most systems. The initial setting is 25.
#TPFDBSW	The size of the SW00SR block. The initial setting is L4 (4095 bytes), the smallest SW00SR block allowed. A 4095 byte block allows as many as five files to be opened without the TPFDF product having to get another block. If there are programs in your system which will have more than five files open at the same time, consider a larger block size to avoid the performance overhead of the GETCC for the additional block. If you change the #TPFDBSW value, make sure the size is defined in the ALCS macro **RECSIZE** and **CISIZE** parameters (see “Modifying the ALCS Generation Macro” on page 25), the BLKSZ macro as described in “Updating the BLKS Macro” on page 23, and the SCTGEN macro SUSIZE and NBRSU parameter values. The default setting of SUSIZE (12-KB) allows the application to use 8KB and the SW00SR to use 4KB. Increasing the SW00SR value decreases the bytes that are available for the application.

Table 10. Setting Symbols in the ACPDBE Macro (continued)

Symbol	Description and settings
#TPFDBGD	General data set support tag names (do not change).
#DBDCNBR	Maximum IDs for data collection. The initial setting is 1300.
#DBCOL1	Size of the record ID index for data collection. The initial setting is ((X'10000' − #TPFDBID) × 4). Do not change this value.
#DBCOL2	Size of the data area for data collection. The initial setting is (#DBDCNBR × 236). Do not change this value.
#DBCOLL	Used to calculate the space required for data collection. The initial setting is the sum of #DBCOL1 and #DBCOL2, rounded up to a 4-KB boundary; that is, (((#DBCOL1 + #DBCOL2 + 4095) ÷ 4096) × 4096). Do not change this value.
#DBDET_MAX	Maximum number of detached core blocks for all open files in the range 20–256 for an ECB. Blocks exceeding this number use short-term pool files. This number can be increased or decreased depending on system resources. The initial setting is 200.
#TO_RELEASE	Percentage of unmodified detached core blocks in the range 1–100 that are released by the system maintenance routine before a pending buffered block in detac mode is moved to a short-term pool file. The default is 10%. The number of unmodified detached blocks is specified by the #DBDET_MAX set symbol.
#TPFDBG2	Does not apply to ALCS systems.
#TPFDBG3	Does not apply to ALCS systems.
#DB2LUFB	The initial setting is OA (do not change).
#DB3LUFB	The initial setting is OAEA (do not change).
#DB4LUFB	The initial setting is OAEX (do not change).
#DF_MAX_DSP	Maximum number of lines that are displayed by the DBDSP macro or `dfdsp` function. The initial setting is 0, indicating that there is no limit.
#DBENUFB	The initial setting is the character on your keyboard that is equivalent to X'4F'.
#DFUTI01	Creating ECBs in TPFDF recoup and the TPFDF capture/restore utility, information and statistics environment (CRUISE) recalculates the distribution of ECBs over the different levels for every so many ordinals as specified. The #DFUTI01 equate determines how often this is done. The initial setting is 500.
#DFUTI02	Creating ECBs in TPFDF recoup and TPFDF CRUISE recalculates the distribution of ECBs over the different levels depending on the ratio of overflows to prime blocks. The #DFUTI02 equate defines this ratio. The initial setting is 50.
#TPFDBD0 to #TPFDBDF	Specifies the data levels to which open files are attached internally. The TPFDF product is shipped to use 15 data levels by default with equates that are sequential from #TPFDBD1 to #TPFDBDF, and then wrapping to #TPFDBD0. **Note:** There is no reason to change the default values. Because the application data levels are preserved across TPFDF calls, the equates will only affect internal data level use in the TPFDF product.
#DBUIO80 to #DBUIO85	Used to support special scrolling. If your UIO and UIS programs support special scrolling, TPFDF programs use these symbols to set the UI2MG2 field in the UIO parameter block to UI2PF.
#DBUIOC8	If your system versions of UIO and UIS do not support scrolling, set this symbol to X'C8'.

Setting Parameters in the C$ACPDBE Header File

The C$ACPDBE header file defines the value for the _TPFDBID symbol.

Table 11. Setting Symbols In The C$ACPDBE Header File

Symbol	Description and settings
_TPFDBID	Specifies the lowest record ID that can be used for TPFDF files. For example, if _TPFDBID is set to X'8000', record IDs X'8000'–X'FFFF' can be used. The initial setting is X'0000' for ALCS environments. **Note:** The lower the value of _TPFDBID, the higher the amount of memory required.

Setting Parameters in the C$CRUUSR Header File

The C$CRUUSR header file contains user-specific settings that you must set before you compile the capture/restore utility, information and statistics environment (CRUISE). The C$CRUUSR header file also defines parameter table default values. These default values are a primary set of values that you can modify when CRUISE is installed. After you have set these values, see “Creating a Parameter Table Index” on page 36 for more information.

Table 12. Parameter Table Defaults

Parameter	Description and settings
CRU_DEF_ECB	The entry control block (ECB) start value as specified by a percent in the range 1–100. The initial setting is 10%.
CRU_DEF_IMB	Indicates if combinations used with parameters WID and ADR select embedded references. The initial setting is Y. **Y** YES **N** NO
CRU_DEF_NLM	Specifies the number of messages to log. The initial setting is −1. **−1** specifies all messages are logged. *log* specifies the number of messages to be logged, where *log* is a number in the range 0–100.
CRU_DEF_NPM	Specifies the number of messages to print. The initial setting is −1. **−1** specifies that all messages are to be printed. *prt* specifies the number of messages to be printed, where *prt* is a number in the range 0–100.
CRU_DEF_RST	Specifies the restore option. The initial setting is R. **N** specifies that the captured fixed and pool files are restored to new pool file addresses where the data structure is rebuilt. **O** specifies that the database is restored to the same fixed and pool file addresses that were captured. **R** specifies that the captured fixed files are restored to the same ordinals that were captured. The captured pool files are restored to new pool file addresses and the database structure is rebuilt. The old pool file addresses are not released when the structure is rebuilt. **Z** specifies that the captured fixed files are restored to the same ordinals that were captured. The captured pool files are restored to new pool file addresses and the database structure is rebuilt. The old pool file addresses are released when the data structure is rebuilt. **X** specifies that the captured fixed and pool files are restored to new pool file addresses and the data structure is not rebuilt.

Table 12. Parameter Table Defaults (continued)

Parameter	Description and settings
CRU_DEF_STA	Specifies if statistics are generated. The initial setting is N. **Y** YES **N** NO
CRU_DEF_SET	Indicates if all pool addresses found are set in use in the pool directory. The initial setting is N. **Y** YES **N** NO
CRU_DEF_TAP	Specifies the default tape name. The initial setting is BFA.
CRU_DEF_TAR	Specifies the target system for a capture. The initial setting is T. **T** TPF **M** MVS
CRU_LOD_V	Specifies the amount of entries that must be available in theALCS environment to run CRUISE. The initial and minimum setting is 5.
CRU_PFA_X_REF	Indicates if all CRUISE functions create an IRCFDF cross-reference prime file while chain chasing. The initial setting is 0. **1** YES **0** NO

Updating the BLKS Macro

The block size (BLKSZ) macro contains information about storage block and record sizes. It is shipped with L0–L4 predefined. Do not change this information. Update the macro to provide additional block types (L5–L8) that your ALCS environment supports. For example, Figure 1 on page 24 shows the BLKSZ macro with a user-defined block type of L5.

```
&BT(1)   SETC  'L0'              ONLY VALID FOR MODE 0
&BS(1)   SETA  128
&BN(1)   SETA  0                 NAB NOT VALID FOR L0 BLOCK
&BC(1)   SETA  0
&BT(2)   SETC  'L1'
&BS(2)   SETA  381
&BN(2)   SETA  &BS(2)-36
&BC(2)   SETA  0*16
&BT(3)   SETC  'L2'
&BS(3)   SETA  1055
&BN(3)   SETA  &BS(3)-36
&BC(3)   SETA  1*16
&BT(4)   SETC  'L3'
&BS(4)   SETA  4000
&BN(4)   SETA  &BS(4)-36
&BC(4)   SETA  2*16
&BT(5)   SETC  'L4'
&BS(5)   SETA  4095
&BN(5)   SETA  &BS(5)-36
&BC(5)   SETA  3*16
&BT(6)   SETC  'L5'              BLOCK TYPE CODE
&BS(6)   SETA  8192              BLOCK SIZE
&BN(6)   SETA  &BS(6)-36         MAXIMUM NAB
&BC(6)   SETA  4*16              CTL BITS
⋮
```

Figure 1. Block Sizes for ALCS

For each additional block that is defined, specify the following:

- **Block type code** (L5–L8)
- **Block size** (in bytes)
- **Maximum next available byte (NAB)**. The maximum NAB is 36 bytes less than the block size.
- **CTL bits**, as follows:

L5	Set to 4*16
L6	Set to 5*16
L7	Set to 6*16
L8	Set to 7*16

Modifying the ALCS Environment for the TPFDF Product

This section describes the adjustments that you must make to your ALCS environment when installing the TPFDF product in an ALCS environment.

Follow the instructions in this section to:

- Modify the ALCS generation macro
- Modify the DXCURID macro
- Modify the SCTGEN macro
- Modify the SEQGEN macro
- Modify the SYSEQ macro
- Modify the DBGEN macro
- Run ALCS generations
- Allocate fixed record data sets
- Install the BDFUACM1 USERMOD
- Change the program configuration table
- Modify the DXCPLIB concatenation

- Allow ALCS recoup to use DBDEF macro statements.

Modifying the ALCS Generation Macro

In the ALCS generation macro, do the following:

- Add TPFDF macro library names to the SOURCE parameter.
- Define L4 (4095 byte) storage blocks and records, required by TPFDF, using the RECSIZE and CISIZE parameters.

Figure 2 shows an example with the required information highlighted. Member BDFGW0 in the sample JCL that is shipped with the TPFDF product contains sample definitions. For more information about the ALCS macro, see *ALCS Installation and Customization*.

Note: For maintainability, it is recommended that you code only one ALCS macro and imbed it in the SCTGEN, SEQGEN, and DBGEN macros.

```
ALCS  ID=W,VERSION=0,                                          -
      SOURCE=('user.maclib',                                   -
      'BDF.V1R1M3.BDFMAC1',                                    -
      'DXC.V2R1M1.DXCMAC1',                                    -
      'DXC.V2R1M1.DXCMAC3',                                    -
      'DXC.V2R1M1.DXCMAC2',                                    -
      'BDF.V1R1M3.BDFMAC2',                                    -
      'BDF.V1R1M3.BDFMAC3'),                                   -
      LOAD='DXC.V2R1M1.DXCLMD4',                               -
      CISIZE=(512,2048,4096,4608),                             -
      RECSIZE=(381,1055,4000,4095),                            -
      PROC=((ASMHC,C),(LKED,LKED))
```

Figure 2. Sample ALCS Macro

Note: In Figure 2 *user.maclib* is the library containing copies of DXCURID and SYSEQ modified with the definitions required by your applications and TPFDF.

Modifying the DXCURID Macro

In the DXCURID macro, define fixed file types.

Figure 3 on page 26 shows an example of the required information. Member DXCURID in the sample JCL that is shipped with the TPFDF product contains sample definitions. For more information about the DXCURID macro, see *ALCS Installation and Customization*.

```
&DXCFN(61) SETC  '#IDFCS'         MISCELLANEOUS SMALL
&DXCFV(61) SETA  56
&DXCFN(62) SETC  '#IDFCL'         MISCELLANEOUS LARGE
&DXCFV(62) SETA  57
&DXCFN(63) SETC  '#IDFC4'         MISCELLANEOUS 4K
&DXCFV(63) SETA  58
&DXCFN(64) SETC  '#IDFUS'         MISCELLANEOUS SMALL
&DXCFV(64) SETA  59
&DXCFN(65) SETC  '#IDFUL'         MISCELLANEOUS LARGE
&DXCFV(65) SETA  60
&DXCFN(66) SETC  '#IDFU4'         MISCELLANEOUS 4K
&DXCFV(66) SETA  61
&DXCFN(67) SETC  '#GR3MSR'        MACRO LABEL SUPPORT
&DXCFV(67) SETA  62
&DXCFN(68) SETC  '#IRDIDF'        DATA COLLECTION
&DXCFV(68) SETA  63
&DXCFN(69) SETC  '#IR02DF'        TPFDF EDUCATAION FILE
&DXCFV(69) SETA  64
&DXCFN(70) SETC  '#IR03DF'        TPFDF EDUCATION FILE
&DXCFV(70) SETA  65
&DXCFN(71) SETC  '#IRCBDF'        CRUISE
&DXCFV(71) SETA  66
&DXCFN(72) SETC  '#IRCFDF'        CRUISE
&DXCFV(72) SETA  67
&DXCFN(73) SETC  '#IRCGDF'        CRUISE
&DXCFV(73) SETA  68
&DXCFN(74) SETC  '#IRCHDF'        CRUISE
&DXCFV(74) SETA  69
&DXCFN(75) SETC  '#IRCIDF'        CRUISE
&DXCFV(75) SETA  70
&DXCFN(76) SETC  '#IRCJDF'        CRUISE
&DXCFV(76) SETA  71
&DXCFN(77) SETC  '#IRCKDF'        CRUISE
&DXCFV(77) SETA  72
```

Figure 3. Sample Fixed File Definitions

Note: #IR02DF and **#IR03DF** are only used with the TPFDF education package.

Modifying the SCTGEN Macro

The SCTGEN macro contains the system control parameters in an ALCS environment and generates the ALCS configuration table. The following parameters are part of the SCTGEN macro that can be specified when the TPFDF product is used in an ALCS environment.

AMODE31=FORCE
specifies 31-bit addressing mode in an ALCS environment. You must be in 31-bit addressing mode to issue a TPFDF call. If your system does not default to 31-bit addressing mode, an application program that issues TPFDF macro calls must be set to 31-bit addressing. You can use the AMODE parameter with the BEGIN macro to specify 31-bit processing for an application program.

NBRSU
specifies the number of type-2 storage units (SUs). Type-2 SUs are required for high-level language application programs run in an ALCS environment.

SUSIZE
specifies the size of type-2 storage units (SUs). Type-2 SUs are required for high-level language application programs run in an ALCS environment. Specify a type-2 SU size of 508- KB for TPFDF C programs compiled using OS/390 LE support.

TPFDF=YES
specifies that the TPFDF product is installed.

Note: If you specify TPFDF=YES, you must include the TPFDF macro libraries in the SOURCE parameter of the ALCS generation macro.

ids
is the maximum number of record IDs for TPFDF data collection. The default is 5000.

STATIC
specifies that a fast-link table is created during ALCS restart, which contains the storage address of each TPFDF system program. If new versions of TPFDF system programs are loaded the fast-link table is not changed. Each entry control block (ECB) that is subsequently created will access the original version of the TPFDF system program. This parameter is the default.

DYNAMIC
specifies that a fast-link table is created for each ECB. The storage address of each TPFDF system program is determined on the first TPFDF call of that program and placed in the fast-link table. If new versions of the TPFDF system programs are loaded, the fast-link table of each ECB that is subsequently created will contain the storage address of the new version.

Note: The DYNAMIC parameter can impact system performance because additional processing is required.

NOTRACE
specifies that TPFDF calls issued by ALCS application programs are not traced. This parameter is the default.

TRACE
specifies that TPFDF calls issued by ALCS application programs are traced. Use this parameter to add TPFDF macro trace data to ALCS dumps and for ALCS conversational and diagnostic data.

Notes:

1. You can alternate between the NOTRACE and TRACE parameters online using the ZTRAC CEP command.
2. Using the TRACE parameter can impact system performance because additional processing is required.

See *ALCS Installation and Customization* for more information about the SCTGEN macro and the ZTRAC CEP command.

Modifying the SEQGEN Macro

In the SEQGEN macro, define the DSNAME parameter with the same name that you use for the LABOUT DD DSN statement in the MLS JCL. The LABOUT DD DSN statement is shown in member MLS in the sample JCL that is shipped with the TPFDF product.

Figure 4 on page 28 shows an example of an SEQGEN macro with the defined DSNAME highlighted.

```
*-------------------------------------------------------------------*
*         FILE REQUIRED FOR TPFDF MACRO LABEL SET SUPPORT           *
*-------------------------------------------------------------------*
         SEQGEN NAME=MLS,                                 -
               TYPE=GENERAL,                              -
               UNIT=(3380,1),                             -
               DISP=(OLD,KEEP,KEEP),                      -
               DSNAME=user.mls_data_set,                  -
               LABEL=(,,,IN,RETPD=0),                     -
               STANDBY=NO,                                -
               VOLCNT=1,                                  -
               BUFNO=1
*-------------------------------------------------------------------*
```

Figure 4. Specifying Macro Label Set Support

Note: In Figure 4 **user.mls_data_set** is the same as the name that you use for the LABOUT DD DSN statement in the MLS JCL.

Modifying the SYSEQ Macro

In the SYSEQ macro, do the following:

- Set the size of your working storage blocks as shown in Figure 5 on page 29.

  The TPFDF product uses symbols to indicate the size (in bytes) of working storage blocks. The #GBSZE symbol must be equated in your SYSEQ to the value shown in Figure 5 on page 29.
- Define record-type ordinals as shown in Figure 5 on page 29.

The sample JCL that is shipped with the TPFDF product contains a sample SYSEQ macro that you can change to suit your installation.

```
         SPACE 1
#GBSZE   EQU   4095                L4 BLOCK SIZE
         SPACE 1
******** ORDINALS FOR #IDFUS
         SPACE 1
#GR31SRF EQU   0
#GR31SRL EQU   #GR31SRF
         SPACE 1
******** ORDINALS FOR #IDFCS
         SPACE 1
#IRDKDFF EQU   2
#IRDKDFL EQU   #IRDKDFF
         SPACE 1
******** ORDINALS FOR #IDFCL
         SPACE 1
#GR32SRF EQU   0
#GR32SRL EQU   #GR32SRF
#IR01DFF EQU   1
#IR01DFL EQU   #IR01DFF
         SPACE 1
******** ORDINALS FOR #IDFU4
         SPACE 1
#GR0VSRF EQU   0
#GR0VSRL EQU   #GR0VSRF
#GR0YSRF EQU   1
#GR0YSRL EQU   #GR0YSRF
#GR27SRF EQU   2
#GR27SRL EQU   #GR27SRF
#GR25SRF EQU   3
#GR25SRL EQU   #GR25SRF
         SPACE 1
******** ORDINALS FOR #IDFC4
         SPACE 1
#GR28SRF EQU   0
#GR28SRL EQU   #GR28SRF
#GR3OSRF EQU   1
#GR3OSRL EQU   #GR3OSRF
#IRA5DFF EQU   5
#IRA5DFL EQU   #IRA5DFF
#IRCADFF EQU   6
#IRCADFL EQU   #IRCADFF
#IRCCDFF EQU   7
#IRCCDFL EQU   #IRCCDFF
#IRCDDFF EQU   8
#IRCDDFL EQU   #IRCDDFF
         SPACE 1
```

Figure 5. Working Storage Sizes and Miscellaneous Record Equates

Note: Ordinal EQUATES are in the form #*xxxxxx*F and #*xxxxxx*L, where F is the first ordinal and L is the last ordinal. The files shown in this figure require only one miscellaneous record, so the first and last ordinals are the same. These ordinals are used when UF1A is assembled.

Modifying the DBGEN Macro

In the DBGEN macro, code USRDTA macros to allocate the records required by the TPFDF product. See Figure 6 on page 30 or member BDFGEND in the sample JCL that is shipped with the TPFDF product for an example of USRDTA macros.

```
USRDTA ACTION=ADD,TYPE=#IDFCS,BAND=4200,                      -
      VFAOPT=FI,SIZE=L1,USAGE=Z1,NUMBER=20
USRDTA ACTION=ADD,TYPE=#IDFCL,BAND=4201,                      -
      VFAOPT=FI,SIZE=L2,USAGE=Z1,NUMBER=20
USRDTA ACTION=ADD,TYPE=#IDFC4,BAND=4202,                      -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=50
USRDTA ACTION=ADD,TYPE=#IDFUS,BAND=4203,                      -
      VFAOPT=FI,SIZE=L1,USAGE=Z1,NUMBER=10
USRDTA ACTION=ADD,TYPE=#IDFUL,BAND=4204,                      -
      VFAOPT=FI,SIZE=L2,USAGE=Z1,NUMBER=10
USRDTA ACTION=ADD,TYPE=#IDFU4,BAND=4205,                      -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=60
USRDTA ACTION=ADD,TYPE=#GR3MSR,BAND=403A,                     -
      VFAOPT=FI,SIZE=L2,USAGE=Z1,NUMBER=99
USRDTA ACTION=ADD,TYPE=#IRDIDF,BAND=4206,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=256
USRDTA ACTION=ADD,TYPE=#IR02DF,BAND=4211,                     -
      VFAOPT=FI,SIZE=L1,USAGE=Z1,NUMBER=26
USRDTA ACTION=ADD,TYPE=#IR03DF,BAND=4212,                     -
      VFAOPT=FI,SIZE=L1,USAGE=Z1,NUMBER=26
USRDTA ACTION=ADD,TYPE=#IRCBDF,BAND=4213,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=200
USRDTA ACTION=ADD,TYPE=#IRCFDF,BAND=4214,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=499
USRDTA ACTION=ADD,TYPE=#IRCGDF,BAND=4215,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=499
USRDTA ACTION=ADD,TYPE=#IRCHDF,BAND=4216,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=32
USRDTA ACTION=ADD,TYPE=#IRCIDF,BAND=4217,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=499
USRDTA ACTION=ADD,TYPE=#IRCJDF,BAND=4218,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=32
USRDTA ACTION=ADD,TYPE=#IRCKDF,BAND=4219,                     -
      VFAOPT=FI,SIZE=L4,USAGE=Z1,NUMBER=200
```

Figure 6. Sample USRDTA Macros

Note: The quantity specified in the NUMBER parameter (in Figure 6) is equal to the quantity of required fixed records (shown in Table 13) plus additional fixed records for future expansion.

Table 13. Fixed Records Required by TPFDF

Name	Size	Quantity
#IDFCS	L1	20
#IDFCL	L2	20
#IDFC4	L4	50
#IDFUS	L1	10
#IDFUL	L2	10
#IDFU4	L4	60
#GR3MSR	L2	99
#IRDIDF	L4	256
#IR02DF	**L1**	**26**
#IR03DF	**L1**	**36**
#IRCBDF	L4	200
#IRCFDF	L4	499
#IRCGDF	L4	499

Table 13. Fixed Records Required by TPFDF (continued)

Name	Size	Quantity
#IRCHDF	L4	32
#IRCIDF	L4	499
#IRCJDF	L4	32
#IRCKDF	L4	200
Note: **#IR02DF** and **#IR03DF** are used for TPFDF education.		

Running ALCS Generations

Run the following generations with the modified data:
- SCTGEN (system generation) macro
- DBGEN (DASD generation) macro
- SEQGEN (sequential file generation) macro.

Allocating Fixed Record Data Sets

Check that the data sets listed in the DASD generation stage 1 listing (under the heading of *Data Base DASD Requirements*) exist and are large enough. If they do not exist or are not large enough, use the sample JCL that is shipped with the TPFDF product to delete, create, and initialize L4 data sets. You will find this JCL in member BDFL4 of the sample JCL.

Installing the BDFUACM1 USERMOD for TPFDF to ALCS

If you are installing TPFDF for the first time, you must install the BDFUACM1 USERMOD for ACM1 to ALCS. The sample JCL that is shipped with the TPFDF product lists sample JCL to receive and apply this USERMOD.

Check the modifications carefully before applying them to your ALCS libraries because you may have to adjust them to suit your system, particularly if you already have USERMODs on the affected programs.

Changing the Program Configuration Table for the TPFDF Product

To change the program configuration table for this version of the TPFDF product, do the following:

1. If there is an entry for a previous shipment of the TPFDF product in your program configuration table (PROGLIST), delete the entry.
2. Add an entry for TPFDF03 in your PROGLIST under the section labeled *LIST OF LOAD MODULES WHICH ARE LOADED DURING ALCS INITIALIZATION*.
3. To use the TPFDF C language interface, add an entry for TPFDFC3 in PROGLIST with the same specifications as TPFDF03.
4. To use TPFDF product code that is written in C language, add an entry for TPFDFH3 in PROGLIST with the same specifications as TPFDF03.

Note: Figure 7 on page 32 shows an example of the required entries.

```
MLIST    DC    0D'0'
            .
            .
            .
         DC    CL8'TPFDFC3 ',AL1(CP0MDUN),AL3(0)  TPFDF 'C' SUPPORT
         DC    CL8'TPFDF03 ',AL1(CP0MDUN),AL3(0)  TPFDF EVERYTHING ELSE
         DC    CL8'UF1A    ',AL1(CP0MDUN),AL3(0)  USER'S UF1A
            .
            .
            .
MLISTE   EQU   *
```

Figure 7. Sample PROGLIST Entries

See *ALCS Installation and Customization* for more information about changing the program configuration table.

Modifying the DXCPLIB Concatenation

1. Add the new BDFLMOD library containing TPFDF03 to the DXCPLIB concatenation in the JCL that you use to IPL your ALCS environment. See member BDFPGLST in the sample JCL that is shipped with the TPFDF product.
2. Assemble segment UF1A, using JCL similar to member BDFALUF1 in the sample JCL that is shipped with the TPFDF product. Assembling UF1A picks up the equates from your SYSEQ and make them available to TPFDF.
3. If no more changes are required, start your ALCS environment per your procedures to activate the modifications and TPFDF. See member BDFRALCS in the sample JCL that is shipped with the TPFDF product.

Allowing ALCS Recoup to Use DBDEF Macro Statements

Sample recoup user exit code, ARD0, ARD1 and ARD2, lets you enter ALCS recoup definitions using DBDEF statements. To use the sample recoup user exit code, perform the following steps:

1. Make sure the ARD0, ARD1, and ARD2 segments are not already in use.
2. Update DBDEF statements to perform recoup. See *TPFDF Database Administration* for more information about the DBDEF macro,
3. Remove the recoup descriptors from the ALCS recoup descriptor record program (BZ90) for files that have recoup functions defined in DBDEF statements.
4. Assemble and load ARD0, ARD1, ARD2, the updated recoup descriptor program (BZ90), and the segments that contain the updated DBDEF statements.
5. Enter **ZUDFM OAI/DBTAB/INIT** to refresh the DBDEF table.

Enabling C Language Support

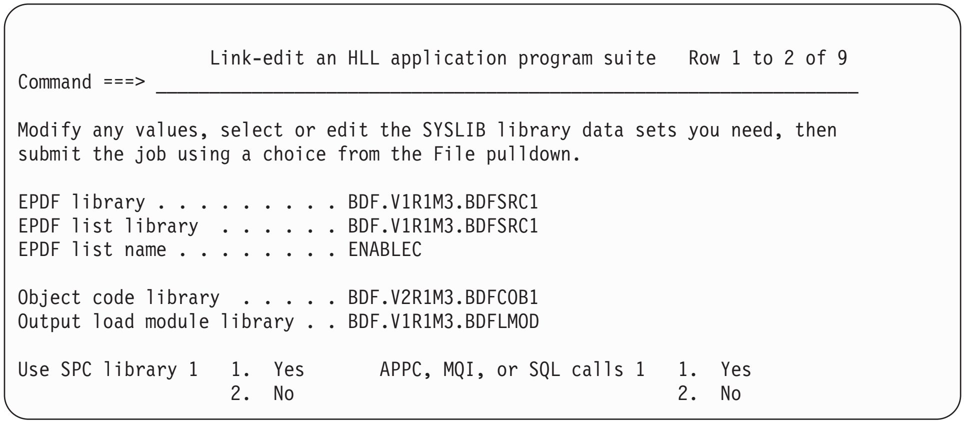
To enable C language support for the ALCS environment, do the following:

1. Ensure that you have C language support installed. See *ALCS Installation and Customization* for more information about C language support in an ALCS environment.
2. Ensure that the program configuration table was updated with the appropriate load modules for C language support. See “Changing the Program Configuration Table for the TPFDF Product” on page 31 for more information about the statements required in the program configuration table.
3. Link-edit the TPFDF C language support code as follows:

a. From the **ALCS primary menu** panel, select action 5 (Application development).

   Note: See *ALCS Installation and Customization* for more information about enabling ALCS ISPF panels.

b. From the **Application development** panel, select action 5 (Link-edit an HLL application suite). This selection causes the following screen to appear:



```
                  Link-edit an HLL application program suite   Row 1 to 2 of 9
Command ===> __________________________________________________________________

Modify any values, select or edit the SYSLIB library data sets you need, then
submit the job using a choice from the File pulldown.

EPDF library . . . . . . . . . BDF.V1R1M3.BDFSRC1
EPDF list library  . . . . . . BDF.V1R1M3.BDFSRC1
EPDF list name . . . . . . . . ENABLEC

Object code library  . . . . . BDF.V2R1M3.BDFCOB1
Output load module library . . BDF.V1R1M3.BDFLMOD

Use SPC library 1   1.  Yes       APPC, MQI, or SQL calls 1   1.  Yes
                    2.  No                                    2.  No
```

The panel fields and the required entries follow:

EPDF library
BDF.V1R1M3.BDFSRC1 is a partitioned data set (PDS) that contains the UFTM segment that enables C functions in an ALCS environment.

EPDF list library
BDF.V1R1M3.BDFSRC1 is a PDS that will contain the ENABLEC file that points to the UFTM segment. You must create the ENABLEC file.

EPDF list name
ENABLEC is a file that you must create that contains only the following item:

```
UFTM
```

Object code library
BDF.V1R1M3.BDFCOB1 is a PDS that contains the TPFDF C object code.

Output load module library
BDF.V1R1M3.BDFLMOD is a PDS that contains the TPFDF load modules.

When you have finished filling in the fields, press the **PF4** key to load (submit) your job to the MVS system.

Activating the TPFDF Product

The following section provides detailed steps to get your TPFDF product operational.

Initializing the TPFDF Files

Attention: Do not initialize the TPFDF files if you have already installed the TPFDF product to this ALCS environment because you will lose the current contents of these files.

Table 14 shows the control files and Table 15 on page 35 shows the noncontrol files that you must initialize if you are installing the TPFDF product to your ALCS environment for the first time.

Table 14. Control Files Required by TPFDF

DSECT Name	File Type	ID	Size	No. of Records	Ordinal Number
GR31SR	#IDFUS	X'FDFF'	L1	1	#GR31SRF
GR0VSR	#IDFU4	X'FDFB'	L4	1	#GR0VSRF
GR0YSR	#IDFU4	X'FDF9'	L4	1	#GR0YSRF
GR27SR	#IDFU4	X'FDF8'	L4	1	#GR27SRF

1. From a CRAS terminal, initialize TPFDF control files.
   a. Initialize GR31SR
   ```
   ZAFIL #IDFUS(0) 000 FDFF0000
   INITIALIZE RECORD ID

   ZAFIL #IDFUS(0) 008 0000000000000000
   CLEAR FORWARD AND BACKWARD CHAINS

   ZAFIL #IDFUS(0) 010 001A000000000000
   INITIALIZE NAB AND FOLLOWING BYTES
   ```
   b. Initialize GR0VSR
   ```
   ZAFIL #IDFU4(0) 000 FDFB0000
   INITIALIZE RECORD ID

   ZAFIL #IDFU4(0) 008 0000000000000000
   CLEAR FORWARD AND BACKWARD CHAINS

   ZAFIL #IDFU4(0) 010 001A000000000000
   INITIALIZE NAB AND FOLLOWING BYTES
   ```
   c. Initialize GR0YSR
   ```
   ZAFIL #IDFU4(1) 000 FDF90000
   INITIALIZE RECORD ID

   ZAFIL #IDFU4(1) 008 0000000000000000
   CLEAR FORWARD AND BACKWARD CHAINS

   ZAFIL #IDFU4(1) 010 001A000000000000
   INITIALIZE NAB AND FOLLOWING BYTES
   ```
   d. Initialize GR27SR
   ```
   ZAFIL #IDFU4(2) 000 FDF80000
   INITIALIZE RECORD ID

   ZAFIL #IDFU4(2) 008 0000000000000000
   CLEAR FORWARD AND BACKWARD CHAINS
    ZAFIL #IDFU4(2) 010 001A000000000000
   INITIALIZE NAB AND FOLLOWING BYTES
   ```
2. Initialize the TPFDF noncontrol files.

   Table 15 on page 35 shows the noncontrol files that you must initialize if you are installing the TPFDF product to your ALCS environment for the first time.

   Enter **ZUDFM OAINIT ID/***xxxx* from a CRAS terminal, where *xxxx* is the ID value shown in Table 15 on page 35. For example, to initialize GR32SR, enter **ZUDFM OAINIT ID/FDFD**.

   Note: Do not initialize files with a file type of POOL.

   See *TPFDF Utilities* for more information about how to use ZUDFM OAINIT.

Table 15. Noncontrol Files Required by TPFDF

DSECT Name	File Type	ID	Size	No. of Records	Ordinal Number/Type
GR25SR	#IDFU4	X'FDF6'	L4	1	#GR25SRF
GR28SR	#IDFC4	X'FDF7'	L4	1	#GR28SRF
GR3MSR	#GR3MSR	X'FDF3'	L2	99	0-98
GR3OSR	#IDFC4	X'FDF5'	L4	1	#GR3OSRF
GR32SR	#IDFCL	X'FDFD'	L2	1	#GR32SRF
IRA5DF	#IDFC4	X'FDA5'	L4	1	#IRA5DFF-#IRA5DFL
IRCADF	#IDFC4	FD1C	L4	1	#IRCADFF-#IRCADFL
IRCBDF	#IRCBDF	FD1D	L4	200	One ordinal per CRUISE ordinal table.
IRCCDF	#IDFC4	FD1E	L4	1	#IRCCDFF-#IRCCDFL
IRCDDF	#IDFC4	FD1F	L4	1	#IRCDDFF-#IRCDDFL
IRCEDF	POOL	FD21	L4	N/A	LT
IRCFDF [1]	#IRCFDF	FD16	L4	499	0–498
IRCGDF [1]	#IRCGDF	FD17	L4	499	0–498
IRCHDF	#IRCHDF	FD18	L4	32	0–31
IRCIDF [1]	#IRCIDF	FD19	L4	499	0–498
IRCJDF	#IRCJDF	FD1A	L4	32	0–31
IRCKDF	#IRCKDF	FD1B	L4	200	One ordinal per CRUISE ordinal table.
IRCMDF[2]	#IRCMDF	FD22	L4	997	0–996 per processor
IRCTDF	POOL	FD23	L4	N/A	LT
IRCXDF	POOL	FD24	L4	N/A	LT
IRDCDF	POOL	X'FD0A'	L4	N/A	LT
IRDIDF	#IRDIDF	X'FDE5'	L4	256	0-255 per (processor+1)
IRDKDF	#IDFCS	X'FDE4'	L1	1	#IRDKDFF
IR01DF	**#IDFCL**	**X'FDE1'**	**L2**	**1**	**#IR01DFF**
IR02DF	**#IR02DF**	**X'F0FC'**	**L1**	**26**	**0-25**
IR03DF	**#IR03DF**	**X'F0FD'**	**L1**	**26**	**0-25**
Note: IR01DF–IR03DF are used for TPFDF education.					
Note: [1] IRCFDF, IRCGDF, and IRCIDF are partitioned files and the ordinals are spread out over the partitions. The number of ordinals listed is the number for each partition. Set the number of partitions in the database definition (DBDEF) macro to the number of loosely coupled processors. For maximum performance, define the number of ordinals as a prime number for each partition.					

Table 15. Noncontrol Files Required by TPFDF (continued)

<table>
<tr><th>DSECT Name</th><th>File Type</th><th>ID</th><th>Size</th><th>No. of Records</th><th>Ordinal Number/Type</th></tr>
<tr><td colspan="6"><b>Note:</b> [2] Use the following formula to calculate the number of ordinals when using CRUISE with MRC support:
<i>#ord = prime (mr / 267 / avgch)</i>

where,
<i>#ord</i> is the ordinal, which is a prime number, to use for the #IRCMDF file.
<i>prime</i> is the function to calculate the closest prime number.
<i>mr</i> is the maximum estimated number of multiple references in a file. You can use the RCI count from the recoup phase 1 activity log as a reference.
<b>267</b> specifies the number of LRECs that are in a data block for the MRC subfile.
<i>avgch</i> is the average number of overflow blocks for each ordinal number in the #IRCMDF database. When using multiple reference check (MRC) support, the recommended number of chains to balance performance and resource (fixed files) requirements is 1.5. The greater the <i>avgch</i> value, the fewer ordinal (fixed files) requirements needed and the more CRUISE capture and restore processing time increases. If you decrease the <i>avgch</i> value, the ordinal (fixed files) requirements needed increases and CRUISE capture and restore processing time decreases. See your database administrator for more information about specifying the <i>avgch</i> value. See <i>TPFDF Utilities</i> for more information about MRC.</td></tr>
</table>

Initializing Command Help

Enter **ZUDFM OAHINIT** to initialize the help facility, which allows you to get online help for commands.

Creating a Parameter Table Index

To create a parameter table index, do the following:

- Enter the ZFCRU EXPAND command to create a parameter table index.
- Enter the ZFCRU DEFAULT command to set the initial default values. See *TPFDF Utilities* for more information about setting default values on a parameter table.

Initializing Database Information Displays

Enter **ZUDFM OAI/DBTAB/REL/CREATE** to initialize the relations database, which allows you to display information about the relationship of files in the database. After the TPFDF product is initialized, enter the command any time you add or change a file on the TPFDF database.

Appendix. What's New

There are no changes.

Index

File Number: S370/30XX-34
Program Number: 5706-196

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.