HITACHI
DATA SYSTEMS

Hitachi Freedom Storage™ Lightning 9900™

Hitachi Multiplatform Data Exchange (HMDE) User's Guide

Notice of Export Controls

Export of technical data contained in this document may require an export license from the United States government and/or the government of Japan. Please contact the Hitachi Data Systems Legal Department for any export compliance questions.

Document Revision Level

Revision	Date	Description
MK-90RD020-P	June 2000	Preliminary Release
MK-90RD020-0	July 2000	Initial Release
MK-90RD020-1	September 2000	Revision 1, supersedes and replaces MK-90RD020-0

Source Document Revision Level

The following source documents were used to produce this revision:

- *Reference Manual File Access Library & File Conversion Utility*, rev 10.1
- ECN number TW-90RD000-0.

Preface

The *Hitachi Multiplatform Data Exchange (HMDE) User's Guide* describes and provides instructions for installing and using the HMDE feature of the Hitachi Lightning 9900™ disk array subsystems. This user's guide assumes that:

- the user has a background in data processing and understands direct-access storage device (DASD) subsystems and their basic functions,
- the user is familiar with the Hitachi Lightning 9900™ array subsystem,
- the user is familiar with the applicable open-system hardware (e.g., SPARCstation, RISC System/6000), operating system (e.g., HP-UX, Solaris, AIX), and application software.
- the user is familiar with the UNIX and/or NT file system, system commands, and utilities.

Note: This document will use the term "9900" to refer to the multiplatform and all-open 9900 disk array subsystems. HMDE is not used on the all-mainframe 9900. Please contact your Hitachi Data Systems account team for the latest information on platform support.

For further information on the 9900 subsystem, please refer to the *Hitachi Lightning 9900™ User and Reference Guide* (MK-90RD008). For further information on Hitachi Data Systems products and services, please contact your Hitachi Data Systems account team, or visit the Hitachi Data Systems worldwide web site at http://www.hds.com.

Note: The use of the HMDE software (FAL/FCU) is governed by the terms of your license agreement with Hitachi Data Systems.

Contents

List of Figures

List of Tables

Chapter 1 Overview of HMDE

1.1 Hitachi Multiplatform Data Exchange

The Hitachi Multiplatform Data Exchange (HMDE) feature enables data stored on the Hitachi Lightning 9900™ subsystems to be converted and transferred between S/390® and open-system platforms and between different open-system platforms. The HMDE mainframe-to-open (HMDEmto) capability enables you to transfer data from S/390 (mainframe) datasets to open-system files. The HMDE open-to-mainframe (HMDEotm) capability enables you to transfer data from open-system files to S/390 datasets. The HMDE open-to-open (HMDEoto) capability enables you to transfer data between open-system platforms without being attached to any S/390 host. HMDE utilizes special HMDE volumes which are dedicated to data exchange operations and are accessed as raw devices to provide the greatest platform flexibility for multiplatform data exchange.

Figure 1.1 illustrates HMDEmto and HMDEotm operations. HMDEmto/otm operations are performed using the File Conversion Utility (FCU) and File Access Library (FAL), which are installed on the open-system host(s). Figure 1.2 illustrates HMDEoto operations, which are performed using the Formatter (FMT) and Allocater (ALC) utilities in addition to FCU and FAL. The FCU program provides the commands and graphical user interface (GUI) for HMDE file transfer operations as well as several important options for data exchange, including EBCDIC-ASCII code conversion and data record padding and delimiters. The FAL is a library of C functions (Visual C++ for Windows NT) which provides an application programming interface for data exchange. The FAL functions can be called by user application programs to read and write data in S/390 datasets on the 9900 subsystem. The FMT and ALC utilities enable the all-open 9900 user to format OPEN-x logical units (LUs) and create intermediate datasets for HMDEoto operations, without having to be attached to any S/390 hosts. OPEN-x is defined as a standard LU type. The 9900 currently supports OPEN-3, OPEN-8, OPEN-9, and OPEN-K devices. Please contact your Hitachi Data Systems account team for the latest information on supported LU types.

HMDE provides the following benefits for the user:

- HMDE provides a centralized data management and disaster recovery environment for both S/390 and open-system data.
- HMDE provides high-speed data transfer over ExSA™ (ESCON®) and fibre channels, freeing up valuable network resources and communication links for application use.
- HMDE's high-speed data exchange enables you to implement file-level backup of open-system data to S/390 storage (e.g., using Harbor).

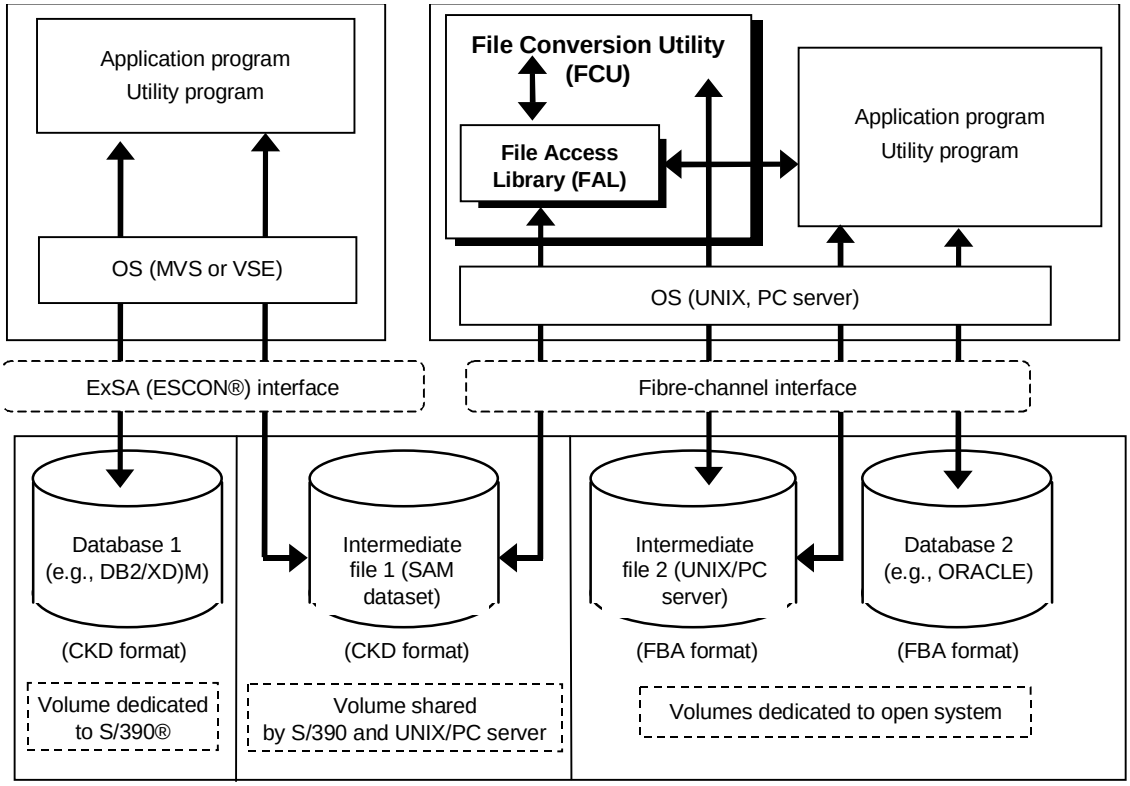


Figure 1.1 HMDEmto and HMDEotm System Configuration

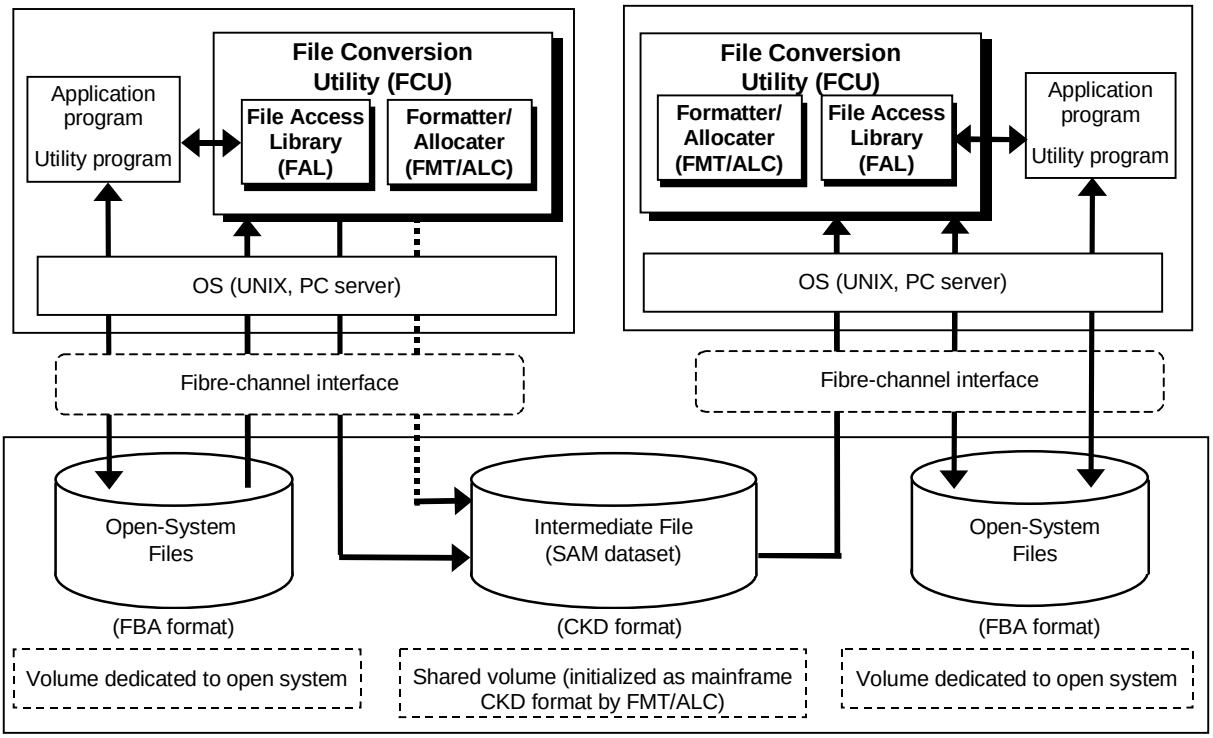


Figure 1.2 HMDEoto System Configuration

1.2 Hitachi Lightning 9900™ Subsystem

The Hitachi Lightning 9900™ subsystem support concurrent attachment to S/390 hosts and multiple open-system platforms, including UNIX-based platforms (e.g., Sun Solaris, HP-UX, IBM AIX) and PC server platforms (e.g., Windows NT, Novell NetWare). The 9900 subsystems provide high-speed response, continuous data availability, scalable connectivity, and expandable capacity for both S/390 and open-system storage. The 9900 subsystem can operate with multihost applications and host clusters, and is designed to handle very large databases as well as data warehousing and data mining applications that store and retrieve terabytes of data.

The Hitachi Lightning 9900™ subsystem provides the following host connectivity options:

- **Multiplatform**. The 9900 subsystem can be configured with fibre-channel (FC) ports in addition to ExSA™ (ESCON®) ports. This type of subsystem is called a multiplatform 9900. HMDE is used on the multiplatform 9900 subsystems.
- **All open**. The 9900 subsystem can be configured with all FC ports (no ExSA™). This type of subsystem is called an all-open 9900. HMDE can be used on the all-open 9900 subsystems for HMDEoto file transfers.
- **All mainframe**. The 9900 subsystem can be configured with all ExSA™ (no FC ports). This type of subsystem is called an all-mainframe 9900. HMDE is not used on the all-mainframe 9900 subsystems.

This document uses the term "9900" to refer to the multiplatform and all-open 9900 subsystems. HMDE is not used on the all-mainframe 9900 subsystems. For further information on the 9900 subsystem, please refer to the *Hitachi Freedom 9900 User and Reference Guide* (MK-90RD008-0). For further information on Hitachi Data Systems products and services, please contact your Hitachi Data Systems account team, or visit the Hitachi Data Systems worldwide web site at http://www.hds.com.

Note: The 9900 subsystem may not support the same open-system platforms and/or operating system (OS) levels. Please contact your Hitachi Data Systems account team for the latest information on open-system platform support.

Note: The Hitachi Multiplatform Backup/Restore (HMBR) feature of the 9900 subsystem enables you to implement S/390-based backup/restore operations (device-level only) for the open-system data stored on the multiplatform 9900. For further information on HMBR, please refer to the *Hitachi Multiplatform Backup/Restore User's Guide* (BO-97DD771), or contact your Hitachi Data Systems account team.

1.3 HMDE Volume Types

The HMDEmto and HMDEotm volumes are S/390 devices which can only be accessed by open-system hosts using the FAL/FCU software. The HMDEoto volumes are open-system devices which cannot be accessed by S/390 hosts. HMDE operations are performed using the following types of HMDE volumes on the 9900 subsystem:

- **HMDEmto**, **HMDEotm**. The HMDE -A volumes (3390-3A and 3380-KA) can be used for HMDEmto and HMDEotm operations. S/390 hosts have normal read/write access to -A volumes [same as 3390-3 or 3380-K logical volume images (LVIs)]. Open-system hosts have read/write access to -A volumes but must use FAL/FCU to access these volumes as raw devices (no mount operation). Figure 1.3 shows the structure of the HMDE -A volumes.

  Note: The -A volumes are not write-protected. Do not execute any open-system write operations to -A volumes (except disk partitioning and labeling). Do not create a file system on an -A volume; this will overwrite the data exchange files on the volume.

- **HMDEmto**. The HMDE -B volumes (3390-3B and 3380-KB) can only be used for HMDEmto operations. S/390 hosts have normal read/write access to -B volumes (same as 3390-3 or 3380-K). Open-system hosts have read-only access to -B volumes and must use FAL/FCU to read these volumes as raw devices (no mount operation). The -B volumes are write-protected from open-system access. The 9900 subsystem will reject all open-system write operations to -B volumes (except disk partitioning and labeling) to protect the S/390 data on these volumes. Figure 1.4 shows the structure of the HMDE -B volumes.

  Note: The open-system host accesses only the volume table of contents (VTOC) area on -B volumes. Catalog or security control functions cannot be used to provide access control for these volumes.

- **HMDEotm**. The HMDE -C volumes (3390-3C and 3380-KC) can only be used for HMDEotm operations. Open-system hosts have read/write access to the -C volumes but must use FAL/FCU to access these volumes as raw devices (no mount operation). S/390 hosts have read-only access to the -C volumes. The 9900 subsystem will reject all S/390 write operations to -C volumes (except VTOC) to protect the open-system data on these volumes. Figure 1.5 shows the structure of the HMDE -C volumes.

- **HMDEoto**. OPEN-x volumes which are formatted with the HMDE Formatter (FMT) utility can only be used for HMDEoto operations. Open-system hosts have read/write access to the OPEN-x FMT volumes but must use FAL/FCU to access these volumes as raw devices (no mount operation). S/390 hosts do not have any access to the OPEN-x FMT volumes. Figure 1.6 shows the structure of the HMDE OPEN-x FMT volumes.

Note for Microsoft Cluster Server:

When installing HMDE devices in a Microsoft Cluster Server (MSCS) environment, you must write signatures on the HMDE volumes before configuring MSCS.

- The MSCS server cannot connect volumes which do not have signatures.
- The volume on which a signature is written cannot be accessed from another server.
- The volume on which a signature is written cannot be shared.
- Only the mainframe and the server which wrote the signature can access the volume which has the signature.

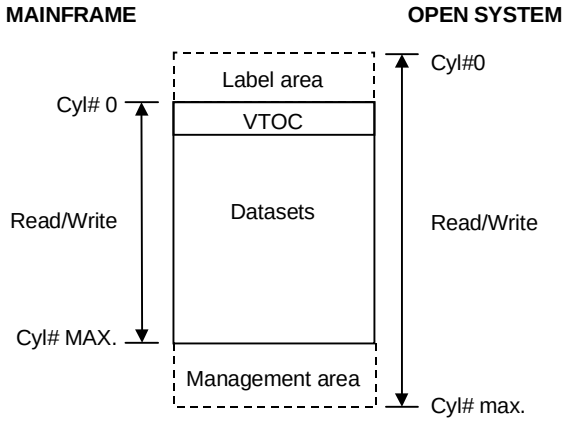


Figure 1.3 3390-3A, 3380-KA Volume Structure (HMDEmto, HMDEotm, HMDEoto)

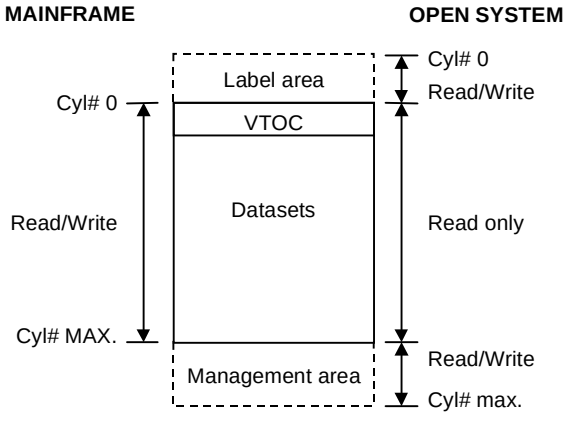


Figure 1.4 3390-3B, 3380-KB Volume Structure (HMDEmto)

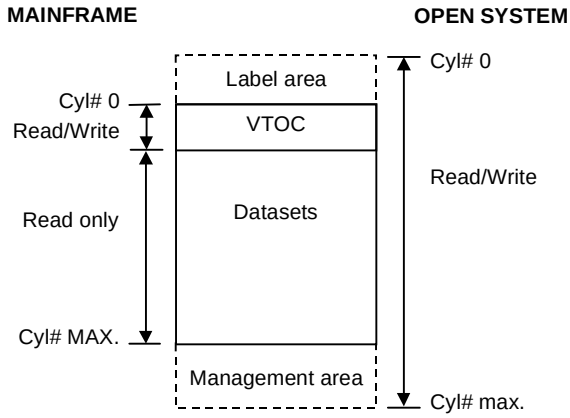


Figure 1.5 3390-3C, 3380-KC Volume Structure (HMDEotm)

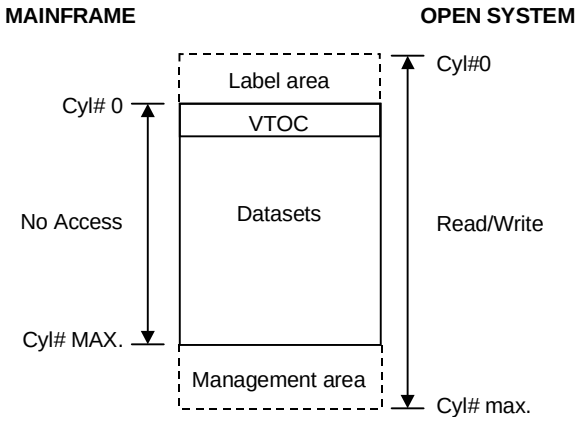


Figure 1.6 OPEN-x FMT Volume Structure (HMDEoto)

Chapter 2 System Requirements and Installation Procedures

2.1 System Requirements

The user should examine existing data exchange needs carefully prior to 9900 subsystem configuration, so that the desired number of HMDE volumes can be installed and configured. Reconfiguring the number of HMDE volumes after 9900 configuration may require reformatting entire array groups, depending on the microcode level of the 9900.

The system requirements for HMDE are:

- FAL/FCU CD for the applicable open-system platform(s). This revision of the *HMDE User's Guide* covers versions 01-01-4x. For earlier versions, please refer to the previous revision of this document.
  - FAL/FCU cannot handle files larger than 2 GB (an error will be returned).
- Multiplatform or all-open 9900 subsystem:
  - HMDEmto: the 9900 must be configured with -B and/or -A HMDE volumes.
  - HMDEotm: the 9900 must be configured with -C and/or -A HMDE volumes.
  - HMDEoto: the user can format OPEN-x LUs as HMDEoto volumes or use -C HMDE volumes. Version 01-01-41 or later of FCU is required to format HMDEoto volumes.
- Remote Console PC and LUN Manager software. LUN Manager enables the user to configure FC ports and create custom-size LUs. If remote LUN Manager is not installed, please contact the Hitachi Data Systems representative about LUN Manager configuration services. For information on 9900 LUN Manager, refer to the *Hitachi Lightning 9900™ LUN Manager User's Guide* (MK-90RD006).
- S/390 operating systems: MVS, VSE (supported by FCU 01-01-40 and later), VOS3
- Open-system platforms and operating system (OS) version level(s):
  HP-UX® 10.2 and 11.0
  Sun® Solaris® 2.5, 2.6, 7 (version 2.5 = 9900 only)
  IBM® AIX® 4.1.x, 4.2.x, 4.3.x (version 4.1 = 9900 only)
  Windows® NT® 4.0 (Workstation or Server)
  Compaq® Tru64® UNIX® (9900 only)
  DIGITAL UNIX 4.0 (9900 only)
  NCR UNIX® SVR4® 3.02 (9900 only, FAL/FCU 01-01-41 and later)
  Sequent® NUMA-Q® DYNIX/ptx® 4.4.4 (9900 only, FAL/FCU 01-01-41 and later)
  Windows 2000®

  Note: The 9900 and 7700E subsystems do not support the same open-system platforms and/or OS levels. Please contact your Hitachi Data Systems account team for the latest information on open-system platform and OS version support.
  - Motif 1.2 (or later) window system software is required for the FCU GUI for UNIX. If Motif is not installed, see Appendix B (*Using FCU Without the GUI*).
  - Superuser (root) login access to the open-system server/workstation is required.

2.2 Installing and Configuring the HMDE Volumes

The HMDE volumes are installed and configured during 9900 subsystem installation and configuration. The HMDE volumes should be dedicated to data exchange operations to avoid accidental overwriting or deletion of important data. The HMDEmto volumes (-B and -A) contain S/390 data to be transferred to open-system LUs. The HMDEotm volumes (-C and -A) contain open-system data to be transferred to S/390 volumes. The HMDEoto volumes contain the intermediate datasets for file transfers between open-system platforms. HMDE does not support concurrent access to HMDE volumes by the S/390 and open-system hosts.

To install and configure the HMDE volumes:

1. Determine exactly how many HMDEmto, HMDEotm, and HMDEoto volumes you will need for your multiplatform data exchange operations. The -A volumes can be used for HMDEmto, HMDEotm, and HMDEoto. The -B volumes are restricted to HMDEmto. The -C volumes are restricted to HMDEotm. The OPEN-x FMT volumes are restricted to HMDEoto. Make sure that the Hitachi Data Systems representative installs the desired number of each type of HMDE volume during 9900 subsystem installation and configuration.

   Note: If you need to change the number of HMDE volumes, please contact your Hitachi Data Systems account team. Reconfiguring the HMDE volumes after 9900 installation may require reformatting entire array groups, depending on the microcode level of the 9900.

2. Complete 9900 installation and device configuration as specified in the 9900 configuration guide for the applicable open-system platform (e.g., *Hitachi Lightning 9900™ HP-UX Configuration Guide*).
   - **Device recognition and device files**. For all open-system platforms, you must verify device recognition and device file creation for all HMDE volumes.
   - **File system/volume group**. Do not create a file system or volume group on any HMDE volume, including the OPEN-x devices which will be formatted for HMDEoto operations. HMDE volumes can only be accessed as raw devices by the open-system host using FAL/FCU (no mount operation).

     Note for Microsoft Cluster Server:
     When installing HMDE devices in a Microsoft Cluster Server (MSCS) environment, you must write signatures on the HMDE volumes before configuring MSCS.
     - The MSCS server cannot connect volumes which do not have signatures.
     - The volume on which a signature is written cannot be accessed from another server.
     - The volume on which a signature is written cannot be shared.
     - Only the mainframe and the server which wrote the signature can access the volume which has the signature.
   - **I/O time-out and I/O queue depth**. Make sure to set the I/O time-out and I/O queue depth values for the HMDE volumes as specified in the applicable 9900 configuration guide.

– **Partition size**. Make sure to specify the correct partition size for the HMDE volumes as specified in the applicable 9900 configuration guide. If the partition size for -A or -B volumes is smaller than the mainframe volume size, the open-system host may not be able to access data to the end of the extent of these volumes.

  For Sun Solaris, use the following partition sizes for the HMDE volumes, and use 2 (two) for the number of alternate cylinders:

Table 2.1 Partition Sizes for Sun Solaris

LVI	Cylinder # for data cylinder extent
3390-3A	0 - 3345
3390-3B	0 - 3339
3390-3C	0 - 3345
3380-KA	0 - 2661
3380-KB	0 - 2655
3380-KC	0 - 2661
OPEN-x	0 - 3335
OPEN-8	0 - 5824
OPEN-K	0 - 2546
OPEN-9	0 - 5824

– **Volume labels**. An HMDE volume with a volume label cannot be shared between open-system platforms which use volume labels. Table 2.2 shows the allowable configurations for sharing HMDE volumes between open-system platforms. HP-UX, IBM AIX and Sequent DYNIX/ptx do not use volume labels, so HMDE volumes can always be shared with these platforms. Labels are optional for DIGITAL UNIX and Windows NT, so HMDE volumes can be shared with these platforms only if they have no label. Sun Solaris and NCR SVR4 always write volume labels, so HMDE volumes can never be shared between these two platforms, but can be shared with the other platforms (HP, IBM, DIGITAL, Sequent, NT) as long as they do not have labels.

Note: Sun Solaris may display the following warning messages when formatting and labeling an HMDE volume. This is normal, and the user can ignore these messages.

Warning: error writing VTOC.
Warning: no backup labels
Write label failed.

3. For UNIX hosts, make sure to set up the desired access privileges for each HMDE volume (e.g., using groups and/or **chmod** command). Please refer to the OS user documentation for information on access permission rights. For Windows NT, Administrator access is required to access the HMDE volumes.

4. On the S/390 host, make sure to initialize and write the VTOC for each HMDEmto and HMDEotm volume to enable the S/390 host to access the volumes. The ICKDSF media maintenance utility can be used to perform these tasks.

5. After FAL/FCU software installation, make sure to format each HMDEoto volume using the HMDE Formatter (FMT) utility on the UNIX/NT host (see section 4.3). This enables the HMDEoto intermediate datasets to be allocated.

Table 2.2 Sharing HMDE Volumes Between Open-System Platforms

		IBM AIX	HP-UX	DIGITAL UNIX	Sequent DYNIX/ptx	Windows NT	Sun Solaris	NCR SVR4
No label	IBM AIX	OK	OK	OK	OK	OK	OK	OK
	HP-UX	OK	OK	OK	OK	OK	OK	OK
	Sequent DYNIX/ptx	OK	OK	OK	OK	OK	OK	OK
Label write option	DIGITAL UNIX	OK	OK	CHK	OK	OK	CHK	CHK
	Windows NT	OK	OK	OK	OK	CHK	CHK	CHK
Label auto-write	Sun Solaris	OK	OK	OK	OK	OK	NO	NO
	NCR SVR4	OK	OK	OK	OK	OK	NO	NO

OK = sharing allowed.
CHK = sharing allowed only if volume has no label.
NO = sharing not allowed.

2.3 Installing the FAL/FCU Software

The FAL/FCU software must be installed on the UNIX/NT server(s) attached to the 9900 subsystem. FAL/FCU software installation for the UNIX-based platforms is different from FAL/FCU installation on Windows NT. The following sections provide instructions for installing FAL/FCU version 01-01-36/38 or later. For instructions on installing versions 01-01-24 and earlier, please refer to the previous revision (-0) of this document.

2.3.1 Installing FAL/FCU on UNIX-Based Platforms

To install the FAL/FCU software on a UNIX-based platform:

1. If FAL/FCU version 01-01-36 or later is already installed (by **cpio** command), you do not need to deinstall it. New installation will overwrite the previous version.
   If FAL/FCU version 01-01-24 or earlier is already installed (by setup program), you must deinstall this older version as follows before installing version 01-01-36 or later:
   a) Log in with the same user ID which was used to install the old FAL/FCU software.
   b) If FAL/FCU was installed from FD using the setup program, see Deinstallation to remove FAL/FCU.
   c) If FAL/FCU was installed from DAT, or if you can't find your FAL/FCU installation FD, move to the directory **fcu/fal.o/dataset.h** and remove FAL/FCU by entering:
      # rm fcu fal.o dataset.h $HOME/FcuMf

      If you cannot find the directory, you can use the following procedure:
      # find / -name "fcu" -print
      # find / -name "fal.o" -print
      # find / -name "dataset.h" -print
      # find / -name "FcuMf" -print
2. Log in to the system as **root**.
3. Insert the FAL/FCU installation media (e.g., CD-ROM) into the drive.
   Note: Verify that the device file for the CD-ROM drive exists. For Sun Solaris, do not use **volcheck** if the CD-ROM device file is not available for auto-mount.
4. Make sure the following six directories exist on the open-system host. If not, create the directories using the **mkdir** command (e.g., **# mkdir /usr/lib/X11/app-defaults**).

/usr	/usr/lib
/usr/bin	/usr/lib/X11
/usr/include	/usr/lib/X11/app-defaults

5. Move to the **root** directory.
6. For UNIX-based systems other than NCR UNIX, copy the FAL/FCU software from the installation CD-ROM as follows:
   # cpio -iBmuv < CD_device_file_name/d

   Note: Use full device file name: wildcard will not work.

   For NCR UNIX, copy and then uncompress the FAL/FCU software as follows:
   # cpio -icmuv /dev/dsk/f03ht/d/

   # uncompress /usr/bin/fcu.Z

7. For Sun Solaris you must set a path to the resource file for each FAL/FCU user:

   a) For C shell, add the following line to the end of the **.cshrc** file in the home directory. If **.cshrc** does not exist, create it and enter the following line:

      setenv XFILESEARCHPATH /usr/lib/X11/app-defaults/%N:$XFILESEARCHPATH

   b) For non-C shell, add the following two lines to the end of the **.dtprofile** file in the home directory. If **.dtprofile** does not exist, create it and enter the following lines:

      XFILESEARCHPATH=/usr/lib/X11/app-defaults/%N:$XFILESEARCHPATH
      export XFILESEARCHPATH

   c) You must log out and log back in to implement these changes.

8. Remove the CD-ROM from the drive.

9. Log out, and then log in again.

Deinstallation. If you need to deinstall FAL/FCU version 01-01-36 or later:

1. Log in to the system as **root**.

2. Remove the FAL/FCU files using the **rm** command as follows, or string the commands:
   # rm /usr/bin/fcu
   # rm /usr/bin/fcunw
   # rm /usr/include/dataset.h
   # rm /usr/lib/libfal.
   # rm /usr/lib/libuoc.
   # rm /usr/lib/X11/app-defaults/FcuMf
   # rm /usr/bin/mfformat
   # rm /usr/bin/allocds

   When deinstalling FAL/FCU version 01-01-41 and later, also remove the following file:
   # rm /usr/bin/listvol

   When deinstalling FAL/FCU version 01-01-45 and later, also remove the following file:
   # rm /usr/lib/libfal.ver

2.3.2 Installing FAL/FCU on Windows NT

To install the FAL/FCU software on a UNIX-based platform:

1. If FAL/FCU is already installed, deinstall it before installing the new version. To deinstall FAL/FCU version 01-01-25 or later:

   a) Open the **Add/Remove Programs** control panel: click on **Start**, click on **Settings**, click on **Control Panel**, then double-click on **Add/Remove Programs**.

   b) Select **FCU** in the list of installed programs, click on the **Add/Remove…** button, and then follow the instructions on screen to complete the deinstallation.

   FAL/FCU versions 01-01-24 and earlier do not support the Windows deinstaller. To deinstall version 01-01-24 or earlier, delete the folder which contains the FAL/FCU software components (**fcu.exe**, **fal.obj**, and **dataset.h**).

2. Insert the FAL/FCU installation CD-ROM into the drive.

3. Run **setup.exe** to start FAL/FCU installation (e.g., double-click on **setup.exe**), and follow the instructions on screen to complete FAL/FCU installation.

2.4 Creating HMDEoto Volumes Using the FMT Utility

After the FAL/FCU software has been installed on the open-system host(s), you can format the HMDEoto volumes using the HMDE Formatter (FMT) utility. This enables you to allocate HMDEoto intermediate datasets. The FMT utility for UNIX is a UNIX command executed from the UNIX command line. The FMT utility for Windows NT is a GUI.

Note: FCU version 01-01-42 or later is required for the FMT utility.

The HMDE FMT utility defines the size of the OPEN-x volume in cylinders. The maximum number of cylinders allowed by FMT are the following:

OPEN-3: 3331

OPEN-8/9: 5818

OPEN-K: 2536

The FMT utility can be used on standard-size OPEN-x volumes and on Custom Volume Size (CVS) volumes. When formatting a CVS OPEN-x LU, use the number of cylinders defined for CVS minus seven (e.g., use 993 cylinders for a CVS device defined with 1000 cylinders). The cylinder size is: one cylinder = 15 tracks, one track = 96 sub-blocks, one sub-block = 512 bytes. Table 2.3 shows the relation between block length and write available capacity per track. The actual data capacity per cylinder = (write available capacity per track) × (15 tracks).

Table 2.3 Relation Between Block Length and Write Available Capacity Per Track

Block Length by Allocater = (A) (Bytes)	Write Available Data per Track (Bytes)	Block Length by Allocater = (A) (Bytes)	Write Available Data Per Track (Bytes)	Block Length by Allocater = (A) (Bytes)	Write Available Data Per Track (Bytes)
23477 - 32760	(A) × 1	1589 - 1684	(A) × 22	565 - 596	(A) × 44
15477 - 23476	(A) × 2	1493 - 1588	(A) × 23	533 - 564	(A) × 45
11477 - 15476	(A) × 3	1397 - 1492	(A) × 24	501 - 532	(A) × 46
9077 - 11476	(A) × 4	1333 - 1396	(A) × 25	469 - 500	(A) × 47
7477 - 9076	(A) × 5	1269 - 1332	(A) × 26	437 - 468	(A) × 48
6357 - 7476	(A) × 6	1205 - 1268	(A) × 27	405 - 436	(A) × 49
5493 - 6356	(A) × 7	1141 - 1204	(A) × 28	373 - 404	(A) × 50
4821 - 5492	(A) × 8	1077 - 1140	(A) × 29	341 - 372	(A) × 51
4277 - 4820	(A) × 9	1045 - 1076	(A) × 30	309 -340	(A) × 52
3861 – 4276	(A) × 10	981 - 1044	(A) × 31	277 - 308	(A) × 53
3477 –3860	(A) × 11	949 - 980	(A) × 32	245 - 276	(A) × 54
3189 – 3476	(A) × 12	917 - 948	(A) × 33	213 - 244	(A) × 55
2933 – 3188	(A) × 13	853 - 916	(A) × 34	181 - 212	(A) × 56
2677 – 2932	(A) × 14	821 - 852	(A) × 35	149 - 180	(A) × 57
2485 – 2676	(A) × 15	789 - 820	(A) × 36	117 - 148	(A) × 58
2325 – 2484	(A) × 16	757 - 788	(A) × 37	85 - 116	(A) × 59

2165 – 2324	(A) × 17	725 - 756	(A) × 38	53 - 84	(A) × 60
2005 – 2164	(A) × 18	693 - 724	(A) × 39	21 - 52	(A) × 61
1877 – 2004	(A) × 19	661 - 692	(A) × 40	1 - 20	(A) × 62
1781 – 1876	(A) × 20	629 - 660	(A) × 41	-	-
1685 – 1780	(A) × 21	597 - 628	(A) × 42	-	-

Note: The write available data per track includes the four-byte RL information and four-byte BL information for each record. When transferring variable-length records, make sure to take this extra required space into account.

WARNING: The HMDE FMT utility erases all data on the OPEN-x LU being formatted. If necessary, back up the data on the OPEN-x LUs prior to FMT formatting.

To format an OPEN-x volume using the HMDE FMT utility for UNIX:

1. Log in to the system as **root**.
2. Enter the following command at the UNIX command line prompt:
   # mfformat -d *devname* -v *VOLSER* [-p *primary_cylinders*]

   -d devname: Specify the raw device name (e.g., /dev/rdsk/c0t1d2 for HP-UX) of the OPEN-x volume being formatted. This parameter is required. Make sure to use the same raw device name for this volume in the HMDEoto volume definition file (see section 4.1).

   -v VOLSER: Specify the VSN of the volume being formatted (A-Z, 0-9, @, #, \). Use only uppercase letters, and do not use any spaces or symbols other than @, #, and \. This parameter is required. Make sure to use the same volser for this volume in the HMDEoto volume definition file.

   -p primary_cylinders: Specify the number of primary cylinders (from decimal 2 through 5818). This parameter is required for custom-size volumes but is optional for standard-size volumes. If this parameter is omitted, the default value of max cylinders is used, specifically: OPEN-3 = 3331, OPEN-8/9 = 5818, OPEN-K = 2536.
3. If the FMT format operation could not be started due to an error condition, the **Format check error** message is displayed. If the FMT format operation did not complete successfully, an error message is displayed. Remove the error condition, and retry the operation (see section C.2 in Appendix C for further information on errors in UNIX).

To format an OPEN-x volume using the HMDE FMT utility for Windows NT:

1. Log in to the system as administrator.
2. Double-click on the **Format** icon to open the Format panel (seeFigure 2.1).
3. On the Format panel, enter the six-character volume serial number for the OPEN-x volume being formatted in the **VOLSER** field. Make sure to use the same volser for this volume in the HMDEoto volume definition file (see section 4.1).
4. Specify the physical drive number (device number) for the OPEN-x volume being formatted in the **Physical drive No** field. Make sure to use the same physical drive number for this volume in the HMDEoto volume definition file.

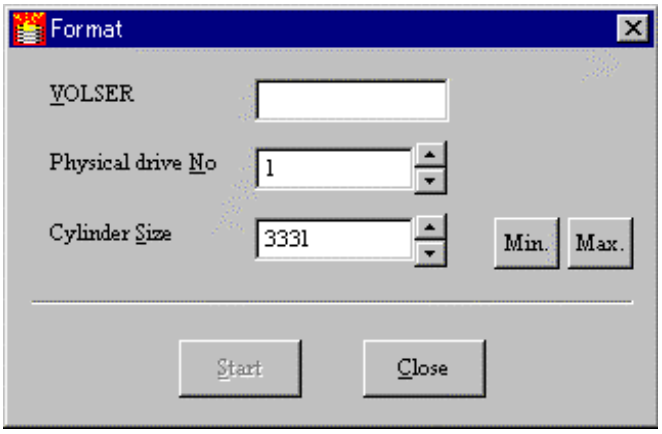


Figure 2.1 FMT Utility for Windows NT

5. Specify the number of cylinders for the OPEN-x volume in the **Cylinder Size** field. The **Min.** button enters 2 (two) cylinders, and the **Max.** button enters 5818 cylinders. If the OPEN-x volume is standard size (e.g., OPEN-3), use the maximum size of 5818 cylinders. If the OPEN-x volume is custom size (e.g., OPEN-3*n CVS device), use the following value: (# of cylinders defined for CVS) – 7. For example, if the CVS OPEN-x volume is defined with 1000 cylinders, enter 993 in the **Cylinder Size** field.

   Note: The maximum size for the Allocater is 4369 cylinders.

6. When the VSN, physical drive number, and cylinder size are correct, select the **Start** button. When the Format confirmation appears (seeFigure 2.2), select **OK** to perform the requested FMT format operation, or select **Cancel** to cancel your request.

7. When the format operation completes successfully, the **Format complete** message is displayed (seeFigure 2.3). If the specified volser has already been used, an error message is displayed to notify you (seeFigure 2.4). If the format operation could not be started due to an error condition, the **Format check error** message is displayed. If the format operation did not complete successfully, one of the following error messages is displayed (see Figure 2.5) (n = system error code):

   Open error! (n) Open process error on specified volume.
   Seek error! (n) Seek process error on specified volume.
   Read error! (n) Read process error on specified volume.
   Write error! (n) Write process error on specified volume.
   Close error! (n) Close process error on specified volume.

8. When you are finished formatting OPEN-x volumes for use as HMDEoto volumes, select **Close** to close the Format panel and exit the FMT utility.

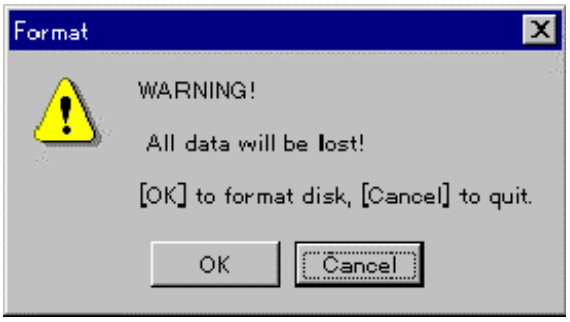


Figure 2.2 FMT Format Warning and Confirmation

Figure 2.3 FMT Format Complete Message

Figure 2.4 FMT VOLSER Used Message

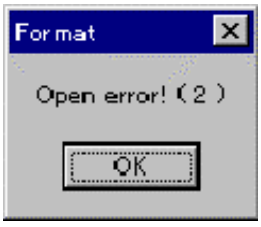


Figure 2.5 FMT Error Message

2.5 Host Access to HMDE Volumes

The user must manage access to the HMDEmto and HMDEotm volumes to prevent illegal I/O access contention between the S/390 and open-system hosts (see section 3.5). These HMDE volumes cannot be accessed concurrently by the S/390 and open-system hosts, and must be varied offline from the S/390 host during HMDE operations. The HMDE volumes should not contain any regularly accessed data and should be dedicated to data exchange operations to avoid accidental overwriting of data.

Note: Do not access the HMDE volume from FAL/FCU when another system (e.g., UNIX, NT, MF) is accessing it. Do not access the HMDE volume from another system when FAL/FCU is accessing it.

Chapter 3 Overview of FCU Operations

3.1 FCU File Transfer Options

For each HMDE operation, FCU requires that the transfer direction (mto or otm) and the source and target files be identified. (An HMDEoto operation consists of one HMDEotm operation followed by one HMDEmto operation.) In addition to these required parameters, FCU provides the following options for HMDE file transfer operations:

Table 3.1 Options for File Transfer Operations

- Code conversion (CC)
- Padding (PAD)
- Delimiters (DEL)
- Empty file (Emp)
- Record description word (RDW)
- VSE record (VSE)

3.1.1 Code Conversion (CC) Option

The code conversion option can be used for HMDEmto and HMDEotm operations. The code conversion option enables you to specify either the default EBCDIC-ASCII code conversion table included with FCU (see Table 3.2), or your own code conversion table (see Table 3.3). When the default table is specified, FCU performs EBCDIC-to-ASCII code conversion for HMDEmto operations and ASCII-to-EBCDIC code conversion for HMDEotm operations as specified in Table 3.2 (see also Appendix D). The user-defined code conversion table must be a binary data file created by placing the target code values in the offset positions which correspond to the source code values.

Always use code conversion when transferring text files between mainframe and open systems. Do not use code conversion when transferring binary data files. Code conversion is available (**EcA** option) but not recommended for HMDEoto file transfers.

Note: FCU versions 01-01-40 and later support the user-defined code conversion table.

Note: The default EBCDIC-ASCII code conversion table is the ACM standard table (not CACM). Appendix D provides the code conversion information for the default table which is shown in Table 3.2. If the default code conversion table does not yield the desired results, create your own code conversion table. Please refer to the IBM code tables for detailed information on EBCDIC-ASCII code conversion.

Table 3.2 Default EBCDIC-ASCII Code Conversion Table for FCU

H / L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL (00)	DLE (10)	DS (80)	(90)	SP (20)	& (26)	- (2D)	(BA)	(C3)	(CA)	(D1)	(D8)	{ (7B)	} (7D)	\ (5C)	0 (30)
1	SOH (01)	DC1 (11)	SOS (81)	(91)	(A0)	(A9)	/ (2F)	(BB)	a (61)	j (6A)	(E5)	(D9)	A (41)	J (4A)	(9F)	1 (31)
2	STX (02)	DC2 (12)	FS (82)	SYN (16)	(A1)	(AA)	(B2)	(BC)	b (62)	k (6B)	s (73)	(DA)	B (42)	K (4B)	S (53)	2 (32)
3	ETX (03)	DC3 (13)	(83)	(93)	(A2)	(AB)	(B3)	(BD)	c (63)	l (6C)	t (74)	(DB)	C (43)	L (4C)	T (54)	3 (33)
4	PF (9C)	TM (9D)	BYP (84)	PN (94)	(A3)	(AC)	(B4)	(BE)	d (64)	m (6D)	u (75)	(DC)	D (44)	M (4D)	U (55)	4 (34)
5	HT (09)	(85)	LF (0A)	RS (95)	(A4)	(AD)	(B5)	(BF)	e (65)	n (6E)	v (76)	(DD)	E (45)	N (4E)	V (56)	5 (35)
6	LC (86)	BS (08)	ETB (17)	UC (96)	(A5)	(AE)	(B6)	(C0)	f (66)	o (6F)	w (77)	(DE)	F (46)	O (4F)	W (57)	6 (36)
7	DEL (7F)	IL (87)	ESC (1B)	EOT (04)	(A6)	(AF)	(B7)	(C1)	g (67)	p (70)	x (78)	(DF)	G (47)	P (50)	X (58)	7 (37)
8	GE (97)	CAN (18)	(88)	(98)	(A7)	(B0)	(B8)	(C2)	h (68)	q (71)	y (79)	(E0)	H (48)	Q (51)	Y (59)	8 (38)
9	RLF (8D)	EM (19)	(89)	(99)	(A8)	(B1)	(B9)	‘ (60)	i (69)	r (72)	z (7A)	(E1)	I (49)	R (52)	Z (5A)	9 (39)
A	SMM (8E)	CC (92)	SW (8A)	(9A)	(D5)	! (21)	(CB)	: (3A)	(C4)	^ (5E)	(D2)	(E2)	(E8)	(EE)	(F4)	(FA)
B	VT (0B)	CUI (8F)	CUI (8B)	CU3 (9B)	. (2E)	$ (24)	, (2C)	# (23)	(C5)	(CC)	(D3)	(E3)	(E9)	(EF)	(F5)	(FB)
C	FF (0C)	IFS (1C)	(8C)	DC4 (14)	< (3C)	* (2A)	% (25)	@ (40)	(C6)	(CD)	(D4)	(E4)	(EA)	(F0)	(F6)	(FC)
D	CR (0D)	IGS (1D)	ENQ (05)	NAK (15)	((28)	) (29)	_ (5F)	’ (27)	(C7)	(CE)	[(5B)	] (5D)	(EB)	(F1)	(F7)	(FD)
E	SO (0E)	IRS (1E)	ACK (06)	(9E)	+ (2B)	; (3B)	> (3E)	= (3D)	(C8)	(CF)	(D6)	(E6)	(EC)	(F2)	(F8)	(FE)
F	SI (0F)	IUS (1F)	BEL (07)	SUB (1A)	| (7C)	~ (7E)	? (3F)	“ (22)	(C9)	(D0)	(D7)	(E7)	(ED)	(F3)	(F9)	(FF)

Note: Appendix D shows the EBCDIC-ASCII code conversion information for this binary table.

Table 3.3 User-Defined Code Conversion Table

Item	Requirement(s)
Size	256 bytes
Format	Binary data
Code length	One byte (two-byte codes cannot be converted)
File name	The following sequences of characters cannot be used in the file name: EA EcA EkJ No If the file name for the code conversion table contains any of these sequences, FCU will ignore the file and use the default table instead.

Legend for Table 3.1

	Bit Positions	
	Hi	Lo
ASCII	8765	4321
EBCDIC (IBM)	0123	4567

3.1.2 PIPE Function

This function transfers data entries from the mainframe to the application program or the utility program for UNIX systems using a “named pipe”. When this function is used, a mainframe dataset can be transferred to an open system. This is a much faster way to transfer data than the Code Conversion method.

Note: This function is only supported for UNIX systems. It is not supported for Windows NT. It is supported for mainframe to open systems data transfer only. This function requires an application program or a utility program to receive data entries using a named pipe.

- Time-Out Value
  - FCU waits for a “Read Data Entries” status message. A time-out error will be reported if the TIME OUT VALUE is not set appropriately. The TIME OUT VALUE should be set in the WAIT_TIME_VALUE environment variable. The limits are 0~1440 seconds (0 = unlimited). The default value is 0 when the timeout value is undefined.
  - The following examples illustrate the use of the WAIT_TIME_VALUE environment variable.

NOTE: After setting the variable, log out and log in again to establish the variable’s value.

  - EXAMPLE 1: For C shell:
    - Add “setenv WAIT_TIME_VALUE 300” to the file“.cshrc”in the home directory.
    - If “.cshrc” does not exist, create it and add the “setenv” line.
  - EXAMPLE 2: For non-C shell:
    - Add “WAIT_TIME_VALUE=300"
    - Add "export WAIT_TIME_VALUE"
    - These two commands must be added to the file “.dtprofile” in the home directory. If “.dtprofile” does not exist, create it and add the lines.

3.1.3 Padding (PAD) Option

The padding option can be used for HMDEmto with variable-length source datasets and for HMDEotm with fixed-length target datasets. When the padding option is specified for HMDEmto, FCU adds padding to each source data record, so that the length of the each record equals the maximum record length. When padding is specified for HMDEotm, FCU adds padding to each source data entity, so that the length of the each target record equals the record length defined for the target dataset. FCU transfers the data entities with padding to the target file/dataset. FCU cannot extract padding from files or datasets. Sections 3.2 and 3.3 describe HMDEmto and HMDEotm operations with padding.

The type of padding added by FCU depends on whether code conversion was also requested:

- **Padding with code conversion (text files)**. When padding and code conversion are both specified, FCU adds **spaces** to the short data entities as needed.
- **Padding without code conversion (binary data files)**. When padding is specified but code conversion is not, FCU adds **0x00** to the short data entities as
- needed.

Note: If you use HMDEmto with padding, the data cannot be transferred back to the original S/390 dataset (the HMDEotm target dataset will not be compatible with the original dataset). If you use HMDEotm with padding, the delimiter option is required.

3.1.4 Delimiter (DEL) Option

The delimiter option can be used for both HMDEmto and HMDEotm operations and enables variable-length records to be transferred between platforms without losing compatibility with the original dataset. When the delimiter option is specified for HMDEmto, FCU adds the specified delimiter to the end of each data entity in the source file, and then extracts and transfers the data entity with delimiter to the open-system target file. When the delimiter option is specified for HMDEotm, FCU extracts each data entity preceding the specified delimiter and transfers the data entities without delimiters to the target dataset. Sections 3.2 and 3.3 describe HMDE operations with delimiters.

The type and length of the delimiter added (or recognized and extracted) by FCU depends on the open-system platform:

- For UNIX-based platforms, you must specify either a carriage return (CR) or a line feed (LF). The length of this delimiter is one byte.
- For Windows NT, you must specify a CR + LF. The length of this delimiter is two bytes.

  Note: Do not use the delimiter option for HMDEotm if the source file contains the same character(s) as the delimiter but used for a purpose other than delimiting data entities. If you do, FCU will interpret the specified delimiter character(s) as delimiters, which can create a target dataset with corrupt records or generate an error condition.

Note: If you use HMDEmto with delimiter (no padding) for variable-length records, you will be able to transfer the data back to the original S/390 dataset later using HMDEotm.

3.1.5 Empty File (Emp) Option

The empty file option can be used for both HMDEmto and HMDEotm operations. When the empty file option is specified, FCU processes an empty source file instead of returning an error. An empty S/390 dataset is a dataset which has no records or only EOF records. An empty open-system file is a file which has a file size of 0 bytes. When an empty S/390 dataset is processed, the open-system target file size = 0. When an empty open-system file is processed, the target dataset will contain only EOF records.

Note: FCU version 01-01-38 or later for UNIX and FCU version 01-01-36 or later for Windows NT are required for the empty file option.

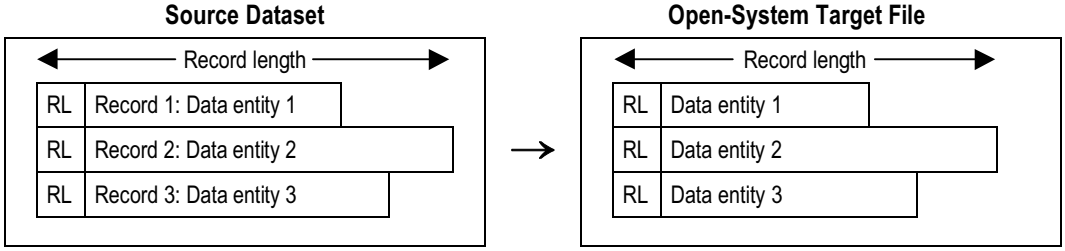
3.1.6 Record Description Word (RDW) Option

The record description word option can only be used for HMDEmto operations on variable-length source datasets. Table 3.4 shows an HMDEmto operation with the RDW option specified. When the RDW option is specified, FCU adds the record description word in binary code to the head of each record in the source dataset, and then transfers the data entity with record length bytes to the open-system target file. The CC, PAD, and DEL parameters must be **No**; if not, FCU returns an error. If the RDW option is specified for a fixed-length source dataset, FCU ignores the RDW option.

Note: If you use HMDEmto with RDW, the data cannot be transferred back to the original S/390 dataset (the HMDEotm target dataset will not be compatible with the original dataset).

Note: FCU version 01-01-38 or later for UNIX and FCU version 01-01-36 or later for Windows NT are required for the RDW option.

Table 3.4 HMDEmto with the RDW Option



3.1.7 VSE Record (VSE) Option

The VSE record option must be used for HMDEmto and HMDEotm operations involving VSE datasets. The VTOC of a VSE dataset does not specify the record format (RF), record length (RL), or block length (BL) of the dataset. The VSE record option enables the user to specify these values so that FCU can process source/target VSE datasets. If you do not specify the VSE record option for a VSE dataset, FCU will return an error. If you specify the VSE record option and the RF, RL, and BL are also specified in the VTOC, FCU will process the dataset if the RF, RL, and BL values are the same, or return an error if the RF, RL, and BL values are not the same. The VSE record option does not apply to ALC-generated intermediate datasets.

Note: FCU version 01-01-40 or later is required for the VSE record option.

3.2 HMDEmto Operations

An HMDEmto operation transfers the data from an S/390 dataset on an HMDE volume to an open-system file on an open-system LU. The object data entities are those contained in all records between the beginning of the file and the end of the file. The end of a dataset is the EOF record or the end of the final extent. The end of an open-system file is the EOF. The HMDEmto source file must be located on an HMDE -B or -A volume on the 9900. If the specified HMDEmto target file does not exist, FCU automatically creates the target file during the HMDEmto operation. If the specified HMDEmto target file already exists, FCU requests confirmation to overwrite the target file (unless the **-nc** option is specified).

The FCU software performs the HMDEmto data transfer operations. FCU version 01-01-40 or later is required for VSE source datasets. FCU version 01-01-41 or later is required for NCR UNIX and Sequent DYNIX/ptx target files. FCU supports both fixed-length and variable-length record formats and provides the following options for HMDEmto data transfer (see section 3.1): code conversion, padding, delimiter, empty file, record description word, and VSE record. The types of HMDEmto operations are:

- HMDEmto with fixed-length record format (see section 3.2.1), and
- HMDEmto with variable-length record format (see section 3.2.2).

Table 3.5 specifies the record format requirements for each type of HMDEmto operation. A fixed-length source dataset can only be transferred to a fixed-length target file, with or without delimiters. Padding cannot be added to a fixed-length source file. A variable-length source dataset can be transferred to a variable-length or fixed-length target file, depending on the padding option, and delimiters can also be added if desired.

Table 3.5 HMDEmto Record Format Requirements

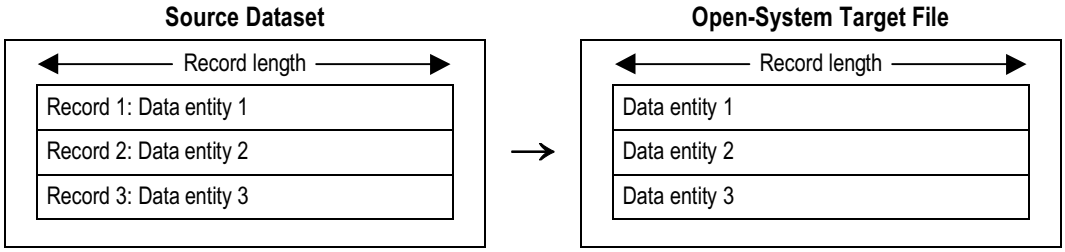
FCU Direction	Padding	Delimiters	Record Format Requirements		See Figure:
			Source Dataset	Target File	
HMDEmto	N/A	No	Fixed-length	Fixed-length	3.2
HMDEmto	N/A	Yes	Fixed-length	Fixed-length	3.3
HMDEmto	No	No	Variable-length	Variable-length	3.4
HMDEmto	Yes	No	Variable-length	Fixed-length	3.5
HMDEmto	No	Yes	Variable-length	Variable-length	3.6
HMDEmto	Yes	Yes	Variable-length	Fixed-length	3.7

3.2.1 HMDEmto with Fixed-Length Record Format

Each fixed-length record in an S/390 dataset includes only the fixed-length data entity. The record length defined for a fixed-length dataset equals the actual length of each data entity. The padding option cannot be used for HMDEmto with fixed-length records.

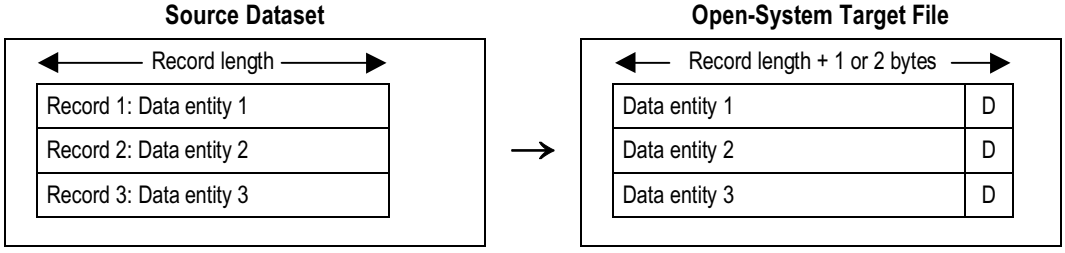
No padding, no delimiters. Table 3.6 shows an HMDEmto operation for a fixed-length source dataset. Padding cannot be added to fixed-length records. FCU extracts and transfers the data entities to the open-system target file. The length of each data entity in the target file equals the record length defined for the source dataset.

Table 3.6 HMDEmto with Fixed-Length Records: No Padding, No Delimiters



With delimiters. Table 3.7 shows an HMDEmto operation with delimiters (D) for a fixed-length source dataset. FCU extracts and transfers the data entities to the open-system target file and adds the requested delimiter to the end of each data entity. The resulting length of each data entity in a UNIX target file equals the original record length plus one byte for the delimiter. The resulting length of each data entity in a Windows NT target file equals the original record length plus two bytes for the delimiter.

Table 3.7 HMDEmto with Fixed-Length Records: Delimiters



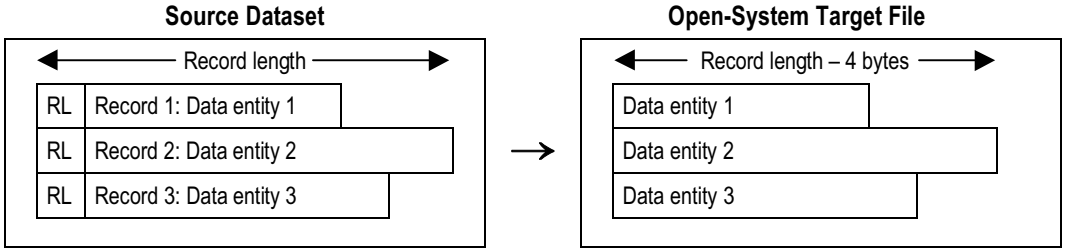
3.2.2 HMDEmto with Variable-Length Record Format

Each variable-length record in an S/390 dataset includes a four-byte RL field and the variable-length data entity. The record length defined for a variable-length dataset equals the maximum allowable record length. ***Note:*** If you want to be able to transfer the data back to the original S/390 dataset later, you must use HMDEmto without padding and with delimiters.

No padding, no delimiters. Table 3.8 shows an HMDEmto operation without padding or delimiters for a variable-length source dataset. FCU extracts and transfers only the data entities to the target file. The RL fields are not transferred. The resulting length of each data entity in the target file is equal to or less than the maximum record length minus four bytes (for the RL field).

Note: If you plan to transfer the data back to the original dataset later using HMDEotm, use HMDEmto with delimiters.

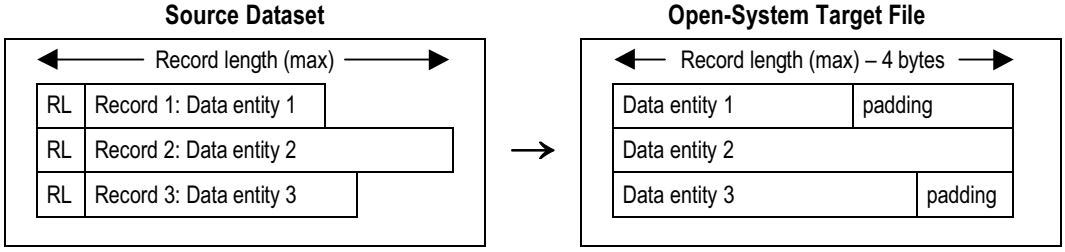
Table 3.8 HMDEmto with Variable-Length Records: No Padding, No Delimiters



With padding. Table 3.9 shows an HMDEmto operation with padding. HMDEmto with padding requires a variable-length source file and produces a fixed-length target file. FCU adds padding to the source records as needed so that the length of each record equals the maximum record length. FCU then extracts and transfers the data entities with padding to the open-system target file. The RL fields are not transferred. The resulting length of each data entity in the target file equals the maximum record length minus four bytes (for the RL field).

Note: If you use HMDEmto with padding, you will not be able to transfer the data back to the original dataset later using HMDEotm.

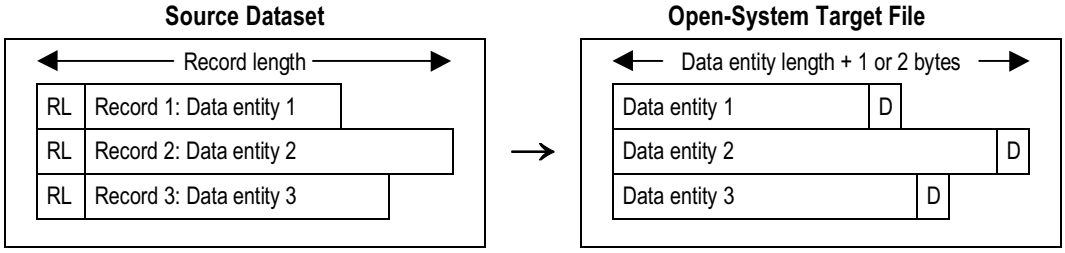
Table 3.9 HMDEmto with Variable-Length Records: Padding



With delimiters. Table 3.10 shows an HMDEmto operation with delimiters (D) for a variable-length source dataset. FCU extracts and transfers the data entities to the open-system target file and adds the requested delimiter to the end of each data entity. The RL fields are not transferred. The resulting length of each data entity in a UNIX target file equals the original data entity length plus one byte for the delimiter. The resulting length of each data entity in a Windows NT target file equals the original data entity length plus two bytes for the delimiter.

Note: If use HMDEmto with delimiters and without padding, you will be able to transfer the variable-length records back to the original dataset later using HMDEotm.

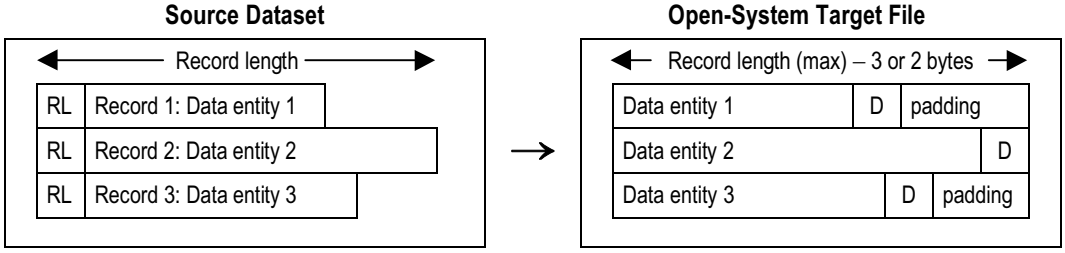
Table 3.10 HMDEmto with Variable-Length Records: Delimiters



With padding and delimiters. Table 3.11 shows an HMDEmto operation with padding and delimiters (D). HMDEmto with padding and delimiters requires a variable-length source file and produces a fixed-length target file. FCU adds the appropriate delimiter to each data entity, adds the appropriate amount of 'padding' so that each record equals the maximum record length, and then extracts and transfers the data entities with padding and delimiters to the open-system target file. The RL fields are not transferred.

Note: If you use HMDEmto with padding and delimiters, you will not to be able to transfer the records back to the original dataset later (the padding cannot be removed).

Table 3.11 HMDEmto with Variable-Length Records: Padding and Delimiters



The resulting length of each data entity in a UNIX target file equals the maximum record length minus three bytes (minus four for the RL, plus one for the delimiter). The resulting length of each data entity in a Windows NT target file equals the maximum record length minus two bytes (minus four for the RL, plus two for the delimiter).

3.3 HMDEotm Operations

An HMDEotm operation transfers the data from an open-system file on an HMDE volume to a target dataset on an S/390 volume. The HMDEotm source file must be located on an HMDE -C or -A volume on the 9900. FCU does not automatically create the HMDEotm target dataset. The target dataset must be created and properly formatted prior to beginning the HMDEotm operation.

The FCU software performs the HMDEotm data transfer operations. FCU version 01-01-40 or later is required for VSE target datasets. FCU version 01-01-41 or later is required for NCR UNIX and Sequent DYNIX/ptx source files and for padding. FCU supports fixed-length and variable-length record formats for HMDEotm operations. FCU provides the following options for HMDEotm operations (see section 3.1): code conversion, padding (01-01-41 and later), delimiter, empty file, and VSE record. The record description word option cannot be used with HMDEotm. FCU automatically extracts delimiters from HMDEotm source files, but cannot add delimiters to HMDEotm source files. FCU can add padding only to variable-length HMDEotm source files. FCU cannot extract padding from HMDEotm source files. The types of HMDEotm operations are:

- HMDEotm with fixed-length record format (see section 3.3.1), and
- HMDEotm with variable-length record format (see section 3.3.2).

Table 3.12 specifies the record format requirements for each type of HMDEotm operation. An open-system source file with fixed-length data entities can only be transferred to a fixed-length target dataset. An open-system source file with variable-length data entities must have delimiters and can be transferred to a variable-length or fixed-length target dataset. If the source file contains padding from a previous HMDEmto transfer operation, the padding is transferred to the target dataset along with the data. If the source file contains delimiters, the delimiters are not transferred to the target dataset.

Note: Do not update the volume that is transferred directly by the HMDEotm.

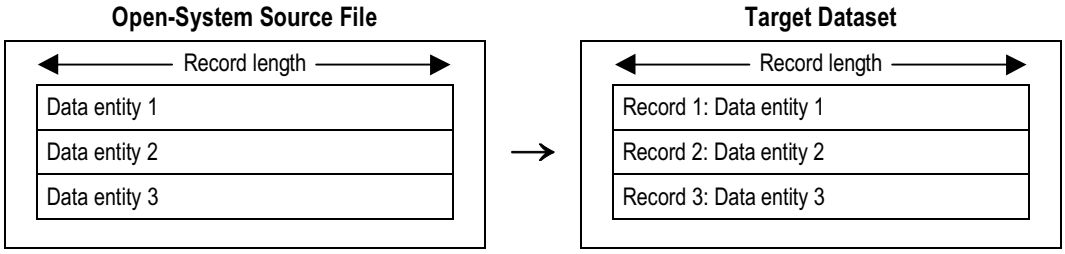
Table 3.12 HMDEotm Record Format Requirements

	Record Format Requirements		
FCU Direction	**Source File**	**Target Dataset**	**See Figure:**
HMDEotm	Fixed-length: no padding, no delimiters	Fixed-length	3.8
HMDEotm	Fixed-length containing padding	Fixed-length	3.9
HMDEotm	Fixed-length containing delimiters	Fixed-length	3.10
HMDEotm	Fixed-length containing padding and delimiters	Fixed-length	3.11
HMDEotm	Variable-length: with delimiters	Variable-length	3.12
HMDEotm	Variable-length: with padding and delimiters	Fixed-length	3.13

3.3.1 HMDEotm with Fixed-Length Record Format

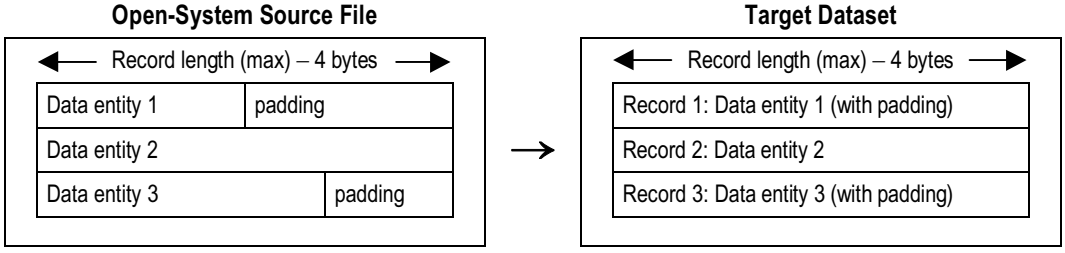
No padding, no delimiters. Table 3.13 shows an HMDEotm operation for a fixed-length source file without padding or delimiters. The target dataset must have fixed-length record format with record length set to the actual length of each data entity. If the data entity length does not exactly match the record length defined for the target dataset, FCU aborts the operation and reports an error.

Table 3.13 HMDEotm with Fixed-Length Records: No Padding, No Delimiters



With padding. Table 3.14 shows an HMDEotm operation for a fixed-length source file with padding from a previous HMDEmto transfer. The original HMDEmto dataset cannot be used as the HMDEotm target dataset. FCU transfers the data entities including padding to the target dataset. The length of each data entity in the source file equals the maximum record length minus four bytes (for the RL field). The target dataset must have fixed-length record format with record length set to the maximum record length minus four bytes. If the length of any record (data entity plus padding) in the source file does not exactly match the record length defined for the target dataset, FCU aborts the operation and reports an error.

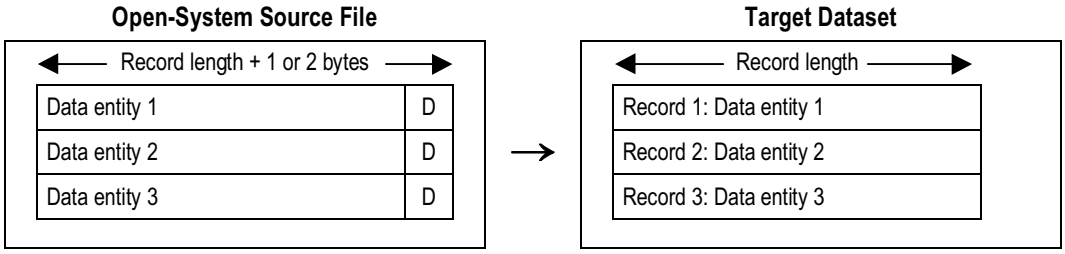
Table 3.14 HMDEotm with Fixed-Length Records: Padding



Note: FCU does not extract padding from HMDEotm source files.

With delimiters. Table 3.15 shows an HMDEotm operation for a fixed-length source file with delimiters from a previous HMDEmto transfer. FCU extracts the data entities from the source file by record length and transfers them to the target dataset. The delimiters are not transferred. The target dataset must have fixed-length record format with record length set to the actual length of each data entity (without delimiter). If the length of any source data entity does not exactly match the record length defined for the target dataset, FCU aborts the operation and reports an error. If the delimiter is not found right after the data entity, FCU aborts the operation reports an error.

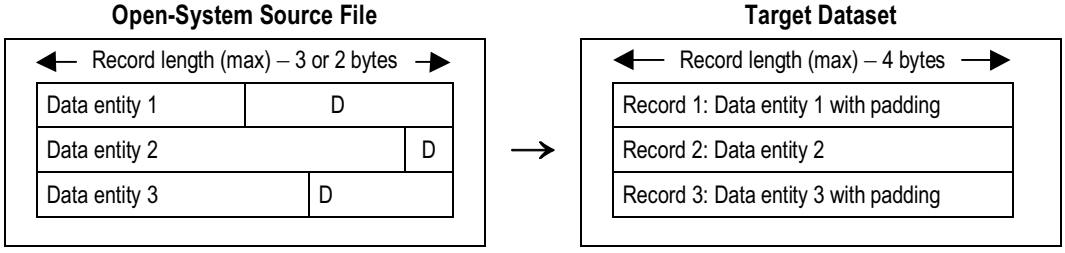
Table 3.15 HMDEotm with Fixed-Length Records: Delimiters



Note: FCU does not add delimiters to HMDEotm source files. If the HMDEotm source file contains delimiters but you specify **No** for the delimiter option, the delimiters will be regarded as part of the data entities and will be transferred to the target dataset.

With padding and delimiters. Table 3.16 shows an HMDEotm operation for a fixed-length source file with padding and delimiters from a previous HMDEmto transfer. FCU removes the delimiters but not the padding and transfers the data entities with padding to the target dataset. The original variable-length dataset cannot be used as the target dataset for this transfer. The target dataset must have fixed-length record format with record length set to the maximum record length minus four bytes. If the length of any source data entity does not match the record length defined for the target dataset, FCU aborts the operation and reports an error.

Table 3.16 HMDEotm with Fixed-Length Records: Padding and Delimiters



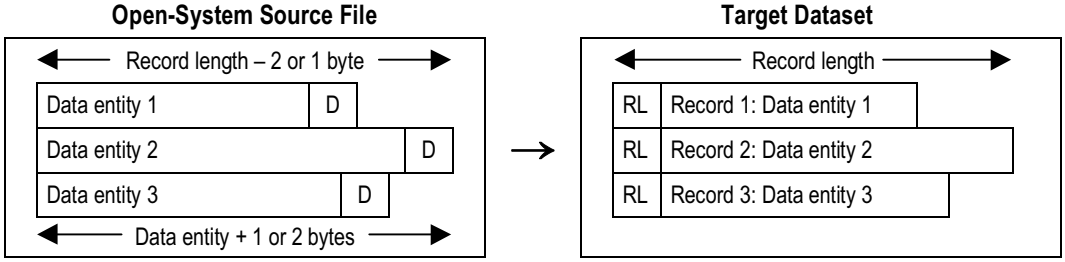
Note: FCU does not extract padding from HMDEotm source files. If the HMDEotm source file contains delimiters but you specify **No** for the delimiter option, the delimiters will be regarded as part of the data entities and will be transferred to the target dataset.

3.3.2 HMDEotm with Variable-Length Record Format

HMDEotm operations can be performed on variable-length source files only if delimiters have already been added to the source file (e.g., from a previous HMDEmto operation). If a variable-length source file without delimiters is processed, FCU will use the maximum record length to construct the target data entities, thereby corrupting the data and rendering the dataset unusable. FCU extracts but does not add delimiters to HMDEotm source files.

With delimiters. Table 3.17 shows an HMDEotm operation for a variable-length source file with delimiters. FCU extracts and transfers the data entities to the target dataset, and automatically adds the four-byte RL field. The delimiters are not transferred. The target dataset must have variable-length record format.

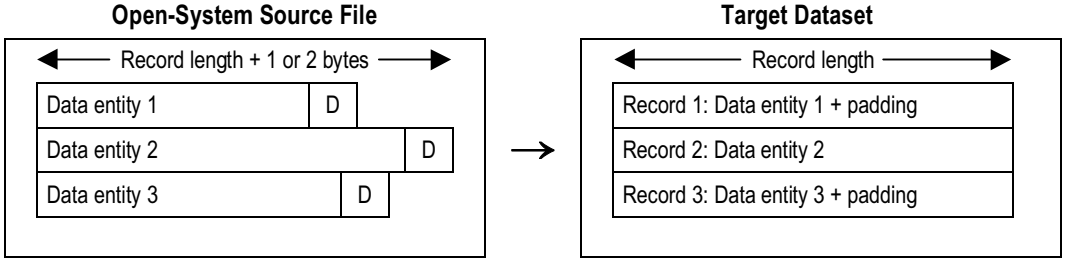
Table 3.17 HMDEotm with Variable-Length Records: Delimiters



If the length of any data entity in a UNIX source file is greater than the maximum record length minus one byte (CR or LF delimiter), FCU aborts the operation and reports an error. If the length of any data entity in a Windows NT source file is greater than the maximum record length minus two bytes (CR+LF delimiter), FCU aborts the operation and reports an error.

With padding and delimiters. Table 3.18 shows an HMDEotm operation with padding for a variable-length source file with delimiters. FCU version 01-01-41 or later is required for HMDEotm with padding added to variable-length source files. FCU adds padding, extracts and transfers the data entities with padding to the target dataset, and automatically adds the four-byte RL field. The delimiters are not transferred. The target dataset must have fixed-length record format with record length defined as needed.

Table 3.18 HMDEotm with Variable-Length Records: Padding and Delimiters



If the length of any data entity in a UNIX source file is greater than the specified record length plus one byte (CR or LF delimiter), FCU aborts the operation and reports an error. If the length of any data entity in a Windows NT source file is greater than the specified record length plus two bytes (CR+LF delimiter), FCU aborts the operation and reports an error.

3.4 HMDEoto Operations

HMDEoto operations transfer data from source files on one open-system platform to target files on another open-system platform. Each HMDEoto file transfer consists of two separate HMDE operations: first an HMDEotm operation transfers the data in the source file to an intermediate dataset, and then an HMDEmto operation transfers the data from the intermediate dataset to the target file. For users with the all-open 9900 subsystem (no attached S/390 host), the intermediate datasets are allocated on OPEN-x FMT volumes. The FMT utility enables you to format OPEN-xLUs (standard or custom size) as HMDEoto volumes (see section 2.4). The ALC utility enables you to allocate intermediate datasets on the OPEN-x FMT volumes (see section 4.4). For users with the multiplatform 9900, the intermediate datasets can be allocated on OPEN-x FMT volumes or on HMDE -A volumes, as desired. When you perform HMDEoto operations which access OPEN-x FMT volumes, the HMDEoto volume definition file must be available for use by FCU (see section 4.1).

Note: FCU version 01-01-41 or later is required for HMDEoto operations using ALC-generated datasets on OPEN-x FMT volumes.

The FCU file transfer options (code conversion, padding, delimiters, etc.) can be used on the HMDEotm and HMDEmto sub-operations as needed.

- Code conversion is not available for HMDEoto transfers.
- Padding can be used but will render the target file incompatible with the source file due to the change in record format from variable-length to fixed-length. If you use padding for the HMDEotm operation, the target file can be transferred back to the same intermediate dataset but not back to the same source file. If you use padding for the HMDEmto operation, the target file cannot be transferred back to the same intermediate dataset or back to the same source file.
- Delimiters can be used to enable bidirectional data transfers. When using delimiters, watch out for files which contain the same character(s) as the delimiter (CR and/or LF) but used for purposes other than delimiting data entities. If you specify the delimiter option for HMDEotm, FCU will interpret all occurrences of the specified delimiter character(s) as delimiters, which can create a dataset with corrupt records or generate an error condition.
- The empty file option can be used to enable empty files to be processed. For example, if a source file specified in your HMDEoto FCU parameter definition file becomes empty, you can add the empty file option to the HMDEotm/mto operations on that file to enable FCU to process the FCU parameter definition file without errors.
- The RDW option is not normally used for HMDEoto operations. If you use the RDW option (HMDEmto operation only), you will not be able to transfer the data back to the same intermediate dataset.
- The VSE record option does not apply to HMDEoto operations which access ALC-generated intermediate datasets on OPEN-x FMT volumes. The only time you would use the VSE option is when transferring a file between open-system platforms via a VSE dataset on a -A HMDE volume. In this case, you must use the VSE record option for both transfers (HMDEotm/mto).

3.5 I/O Access Contention

The HMDE volumes can only be accessed by open-system hosts using the FAL/FCU software. The S/390 hosts have normal read/write access to the -B and -A volumes, read-only access to the -C volumes, and no access at all to the OPEN-x FMT volumes. The open-system hosts have read/write access to the -C, -A, and OPEN-x FMT volumes and read-only access to the -B volumes. The open-system hosts must use FAL/FCU to access all HMDE volumes.

WARNING: Concurrent access to the HMDE volumes by the S/390 and open-system hosts is not supported. The user is responsible for managing access to HMDE volumes to avoid I/O contention between the S/390 and open-system hosts. Since FCU accesses only the VTOC area of the HMDE -B volumes, catalog or security control functions cannot be used to provide access control for the 3390-3B and 3380-KB volumes.

The S/390 host can issue a **reserve** command to reserve a volume for exclusive use. The S/390 **reserve** command prevents access by all other hosts, including all other S/390 hosts and all open-system hosts. The open-system host can also reserve a volume to exclude I/Os issued by other systems. The open-system **reserve** command prevents access by all other open-system hosts, but S/390 hosts still have normal access to HMDEmto and HMDEotm volumes reserved by open-system hosts. These **reserve** commands affect HMDE operations as follows:

- **Reserved by S/390 host**. When an HMDE volume is reserved by the S/390 host, HMDE operations cannot be performed on that volume, because the FAL/FCU access from the open-system host will terminate unsuccessfully. Open-system access other than read or write I/Os can be executed successfully.

  Note: Open-system access to an S/390-reserved volume may complete successfully if the open-system retries the operation after the reserve is released. However, since the time interval before a retry varies depending on the open-system platform and the S/390 application which issued the reserve, the success of retry operations on reserved volumes cannot be guaranteed.

- **Reserved by open-system host**. When an HMDE volume is reserved by the open-system host, HMDE operations can be performed only from the host which reserved the volume. HMDE operations from any other open-system host will terminate unsuccessfully. Open-system reserve does not affect S/390 access to the HMDE volume.

- **Unreserved**. When an HMDE volume is not reserved by any S/390 or open-system host, HMDE operations can be performed from any open-system host using FAL/FCU. All S/390 hosts and all open-system hosts have access to unreserved volumes.

The user should implement exclusive access control and job coordination at the system level for the HMDE volumes. The user should also take the following steps to avoid I/O contention problems for the HMDE volumes:

- **Open-system access**. When the open-system host needs to access an HMDE volume, vary the volume and its channel path offline from all S/390 hosts.

- **S/390 access**. When the S/390 host needs to access an HMDE volume, stop all open-system access to the corresponding LU. For AIX, vary off the volume group(s). For Windows NT, use **unaccess**. Do not use any open-system program which accesses unmounted LUs (e.g., AIX SMIT, HP-UX SAM, NT Disk Administrator).

3.6 Bidirectional Data Transfer

HMDE supports bidirectional data transfer for both fixed-length and variable-length S/390 datasets. Bidirectional data transfer involves transferring data from S/390 datasets to open-system files and then back to the original S/390 datasets again. The requirements for bidirectional data transfer are:

- For all HMDEmto operations, do not specify the record description word (RDW) option. If the RDW option is specified for an HMDEmto data transfer, the subsequent HMDEotm target dataset will not be compatible with the original dataset.
- For HMDEmto with fixed-length datasets, do not specify the delimiter option, since the data entities are extracted by length. If you add delimiters for the HMDEmto transfer, the subsequent HMDEotm target dataset will not be compatible with the original dataset.
- For HMDEmto with variable-length datasets, you must add delimiters but not padding. If delimiters are not added or if padding is added for the HMDEmto transfer, the subsequent HMDEotm target dataset will not be compatible with the original dataset.
- For HMDEotm operations, do not specify the delimiter option if the source file contains the same character(s) as the delimiter (CR and/or LF) but used for purposes other than delimiting data entities. If you specify the delimiter option for HMDEotm, FCU will interpret all occurrences of the specified delimiter character(s) as delimiters, which can create a dataset with corrupt records or generate an error condition.

Chapter 4 Preparing for FCU and FAL Operations

Before you prepare for FCU and FAL operations on the open-system host, make sure that your HMDE environment has been set up and defined correctly, including HMDE volume configuration (see section 2.2), FAL/FCU software installation (see section 2.3), and HMDEoto volume formatting (see section 2.4). After your HMDE environment is set up, you need to perform the following tasks to prepare for FCU and FAL operations:

- Create the HMDE volume definition file(s) (see section 4.1),
- Verify S/390 dataset requirements for HMDEmto and HMDEotm (see section 4.2),
  - ***Note:*** FAL/FCU cannot handle files larger than 2 GB (an error will be returned).
- Allocate the intermediate datasets for HMDEoto (see section 4.3) (not required for FAL operations), and
- Create one or more FCU parameter definition files (if needed) (see section 4.4) (not required for FAL operations).

4.1 Creating the HMDE Volume Definition File(s)

The HMDE volume definition file contains the volume association parameters for the HMDE volumes on the 9900. This file must be created before you can use FCU or FAL to access data on these volumes. The volume association parameters define the HMDE volume by associating the volume serial number (VSN or volser) with the open-system device file for the same logical volume. Table 4.1 describes the HMDE volume association parameters. Figures 4.1 through 4.7 show the structure and contents of the HMDE volume definition file for each supported platform. The numbers in Table 4.1 refer to the numbers in Figure 4.1 - Figure 4.7.

The -A, -B, and -C HMDE volumes and the OPEN-x-HMDEoto volumes can be defined in the same HMDE volume definition file. For example:

```
XXX/XXXXXX MVS01 3390-3A
YYY/YYYYYY VSN01 OPEN-3
end
```

Table 4.1 HMDE Volume Association Parameters

Number	Name	Function	Enter
①	Device file name	Specifies raw device (partition) name defined for open-system.	Character-type device file name (e.g., **c1t0d2** for HP-UX, **c1t0d2s1** for Solaris, **sd3** for DYNIX/ptx).
②	VOLSER	Specifies logical volume defined for S/390.	Six-character volser (e.g., HMDE45). A volser can use the following characters: A-Z, 0-9, @, #, \
③	LVI or LU type (emulation)	Specifies LVI or LU type of HMDE volume.	Correct LVI/LU for HMDE volume: 3390-3A, -3B, -3C, 3380-KA, -KB, -KC, or OPEN-x-HMDEoto. Make sure to define all OPEN-x FMT volumes in a separate file.
④	Carriage return	Marks end of parameter set.	Make sure to press the **Return** key (**Enter** key for Windows NT) at the end of each line.
⑤	End of file	Marks end of parameter file.	**end**

To create the HMDE volume definition file:

1. Open a new empty text file. For UNIX-based systems, use the UNIX vi editor (e.g., **vi datasetmount.dat**). For Windows NT, use any text editor, and make sure to use plain text. The file name must be **datasetmount.dat** (all lowercase), and the file must be located in the current working directory when you start FCU. If you are creating two HMDE volume definition files, use **datasetmount1.dat** and **datasetmount2.dat**, and remove the "1" or "2" from the desired file before starting FCU.
2. Add the volume association parameters for the HMDE volumes to the file (see Figure 4.1 through Figure 4.7).
   - Put at least one space between each parameter, and press the **Return** key at the end of each line to separate the parameter sets. All three parameters (device name, volser, LVI type) are case-sensitive. If you add comments to the file, make sure that each comment line starts with **#**. Make sure to enter **end** on the last line of the file.
3. When you are done adding the volume association parameters for each HMDE volume to the volume definition file, save your changes and exit the text editor.

```
/dev/rdsk/cxtydzsw  volser  3390-3B          ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3390-3A          ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3390-3C          ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KB          ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KA          ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KC          ← Volume association parameters.
       ①            ②      ③    ④          ← See Table 4.1.
 :
end
⑤
```

Note: x = controller number, y = SCSI target ID (TID), z = LUN, w = partition (or slice)

Figure 4.1 HMDE Volume Definition File for Sun Solaris (mto/otm shown)

```
/dev/rdsk/cxtydz  volser OPEN-x          ← Volume association parameters.
/dev/rdsk/cxtydz  volser OPEN-x          ← Volume association parameters.
/dev/rdsk/cxtydz  volser OPEN-x          ← Volume association parameters.
/dev/rdsk/cxtydz  volser OPEN-x          ← Volume association parameters.
       ①          ②      ③     ④       ← See Table 4.1.
 :
end
⑤
```

Note: In cxtydz, x = controller number, y = SCSI TID, z = LUN. In OPEN-x, x = 3, 8, K, or 9.

Figure 4.2 HMDE Volume Definition File for HP-UX (oto shown)

```
/dev/rhdiskn  volser  3390-3B          ← Volume association parameters.
/dev/rhdiskn  volser  3390-3A          ← Volume association parameters.
/dev/rhdiskn  volser  3390-3C          ← Volume association parameters.
/dev/rhdiskn  volser  3380-KA          ← Volume association parameters.
    ①          ②      ③     ④        ← See Table 4.1.
 :
end
⑤
```

Note: n = disk ID number (note that the first, second, and third drives are 0, 1, 2).

Figure 4.3 HMDE Volume Definition File for IBM AIX (mto/otm shown)

```
/dev/rrzXYZ  volser OPEN-x                          ← Volume association parameters.
/dev/rrzXYZ  volser OPEN-x                          ← Volume association parameters.
/dev/rrzXYZ  volser OPEN-x                          ← Volume association parameters.
   ①         ②      ③     ④                       ← See Table 4.1.
 :
end
⑤
```

Note: X = b through h = LUN1 through LUN7 (no letter is used for LUN0); Y = SCSI/fibre bus number × 8 + SCSI TID; Z = partition = a through h. For example, rrzc18a = SCSI TID 2, LUN2 (partition a) on SCSI/fibre bus 2.

Note: OPEN-x = 3, 8, K, or 9

Figure 4.4 HMDE Volume Definition File for DIGITAL UNIX (oto shown)

```
/dev/rdsk/cxtydzsw  volser  3390-3B                 ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3390-3C                 ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KB                 ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KA                 ← Volume association parameters.
/dev/rdsk/cxtydzsw  volser  3380-KC                 ← Volume association parameters.
       ①            ②      ③    ④                 ← See Table 4.1.
 :
end
⑤
```

Note: x = controller number, y = SCSI target ID (TID), z = LUN, w = partition (or slice)

Figure 4.5 HMDE Volume Definition File for NCR UNIX (mto/otm shown)

```
/dev/rdsk/sdx  volser OPEN-x                        ← Volume association parameters.
/dev/rdsk/sdx  volser OPEN-x                        ← Volume association parameters.
/dev/rdsk/sdx  volser OPEN-x                        ← Volume association parameters.
      ①        ②      ③    ④                      ← See Table 4.1.
 :
end
⑤
```

Note: x = device number assigned by OS.

Note: OPEN-x = 3, 8, K, or 9

Figure 4.6 HMDE Volume Definition File for Sequent DYNIX/ptx (oto shown)

```
\\.\PHYSICALDRIVEn  volser  3390-3B                ← Volume association parameters.
\\.\PHYSICALDRIVEn  volser  3390-3A                ← Volume association parameters.
\\.\PHYSICALDRIVEn  volser  3390-3C                ← Volume association parameters.
\\.\PHYSICALDRIVEn  volser  3380-KB                ← Volume association parameters.
       ①            ②      ③    ④                ← See Table 4.1.
 :
end
⑤
```

Note: n = disk ID number.

Figure 4.7 HMDE Volume Definition File for Windows NT (mto/otm shown)

4.2 Verifying S/390 Dataset Requirements

FAL and FCU have specific requirements for the HMDE source and target datasets. Table 4.2 specifies the requirements for HMDE datasets. The FCU GUI (see sections 0 and 5.2) allows the user to display the dataset attributes and verify the dataset requirements. FCU for UNIX (version 01-01-41 and later) also provides the **listvol** function to display S/390 dataset attributes without using the GUI (see section B.2). The HMDEotm target dataset (which can also be an HMDEoto intermediate dataset) must be created and properly configured before the HMDE operation is performed. FCU does not support automatic expansion of the extent during HMDEotm operations. The HMDE ALC utility allocates intermediate datasets in accordance with the requirements specified below.

Table 4.2 S/390 Dataset Requirements

Item	Requirement(s)
Dataset organization (DO) type	SAM (sequential-access method). FAL/FCU does not support any other DO types (e.g., DAM, VSAM, PAM). If a non-SAM dataset is specified, FAL/FCU will return an error. Multiple-volume datasets are not supported. FAL/FCU can only process the portion within one logical volume.
Dataset name	No spaces. If FAL/FCU encounters a space, it will accept the characters before the space as the dataset name and continue processing.
Record format (RF)	Fixed-length or variable-length record format. FAL/FCU does not support undefined-length or spanned record formats. If an illegal RF is detected, FAL/FCU will return an error. No key. If a record with a key is accessed, FAL/FCU will return an error. For HMDEotm, the record format of the target dataset must be preconfigured to match the record format of the data entities in the source file. For VSE source and target datasets, the VSE record option must be used to specify the RF.
Block length (BL)	Any length within the extent supported by the OS. If an illegal BL is detected, FAL/FCU will return an error. For HMDEotm, the block length of the target dataset must be preconfigured to match the block length of the data entities in the source file. For VSE source and target datasets, the VSE record option must be used to specify the BL.
Record length (RL)	Any length within the extent supported by the OS. If an illegal RL is detected, FAL/FCU will return an error. ***Note:*** FAL/FCU cannot process a variable-length dataset which includes a record with no data entity (RL = 4). For HMDEotm, the record length of the target dataset must be preconfigured to match the record length of the data entities in the source file. For VSE source and target datasets, the FCU VSE record option must be used to specify the RL.
Track format	Standard record 0 (R0). FAL/FCU cannot process tracks with nonstandard R0.
VTOC	For MVS: standard or index VTOC. For an index VTOC, FAL/FCU ignores the index and accesses the entire VTOC sequentially. For VSE: the user must specify the RF, BL, and RL using the FCU VSE record option. ***Note:*** The FAL functions cannot be used on VSE datasets. For AIX: define the VTOC and dataset sequentially from the beginning of the volume to ensure optimal processing. If not, FCU processing may take longer than normal.
Database file	Direct access is not supported; must be converted to a SAM file.

4.3 Allocating HMDEoto Intermediate Datasets Using the ALC Utility

When you perform HMDEoto operations using OPEN-x FMT volumes, you must allocate the intermediate datasets before starting the file transfer operations. The HMDE Allocater (ALC) utility can only be used on OPEN-x volumes which have already been formatted using the HMDE FMT utility (see section 2.4). ***Note:*** FCU version 01-01-41 or later is required for the ALC utility.

Note: For versions 01-01-41 and earlier: The ALC utility for UNIX is a UNIX command executed from the UNIX command line. The ALC utility for Windows NT is a GUI. The ALC utility for UNIX can only be used on volumes formatted with the FMT utility for UNIX. The ALC utility for Windows NT can only be used on volumes formatted with the FMT utility for Windows NT.

Note: The capacity of the intermediate dataset varies depending on block length. Use the information in Table 4.2 to calculate the required size for the intermediate dataset. When transferring variable-length records, make sure to take the four-byte RL information and four-byte BL information for each record into account.

UNIX - To allocate an HMDEoto intermediate dataset using the ALC utility:

1. Log in to the system as **root**.
2. Enter the following command at the UNIX command line prompt:
   # allocds -d *devname* [-n *datasetname*] [-f *recform*] [-r *reclen*] [-b *blocklen*] [-c *cylinders*]

   Note: Enter only one value for each parameter. You can only allocate one dataset at a time.

   -d devname: Specify the raw device name of the OPEN-x volume on which the dataset is being allocated. This parameter is required and must be specified.

   -n datasetname: Specify the name of the dataset being allocated (maximum forty-four characters: A-Z, 0-9, @, #, ., \). Use uppercase letters only, and do not use any spaces or symbols other than @, #, ., and \. This parameter is required. If not specified, ALC will return the residual capacity (free space) on the specified volume in number of cylinders.

   -r recform: Specify the record format of the dataset being allocated: **F** (fixed-length and de-blocking), **FB** (fixed and blocking), **V** (variable and de-blocking), or **VB** (variable and blocking). This parameter is required. If not specified, the default value of **F** is used.

   -r reclen: Specify the record length (decimal) of the dataset being allocated: **1 to 32760**. This parameter is required. If not specified, the default value of 4096 is used.

   -b blocklen: Specify the block length (decimal) of the dataset being allocated.
   When record format = F, block length = record length.
   When record format = FB, block length = record length × N (N = integer).
   When record format = V/VB, block length = record length + 4 or more.
   This parameter is required. If not specified, the following default values are used:
   When record format = F/FB, default block length = record length.
   When record format = V/VB, default block length = record length + 4.

   -c cylinders: Specify the size of the dataset being allocated in number of cylinders (decimal). This parameter is required. If not specified, the default value of 100 is used.

3. If the ALC allocate operation could not be started due to an error condition, the **Allocate check error** message is displayed. If the ALC allocate operation did not complete successfully, an error message is displayed. Remove the error condition, and retry the operation. See section C.2 in Appendix C for further information on errors in UNIX.

Windows NT: To allocate an intermediate HMDEoto dataset using the ALC utility:

1. Log in to the system as administrator.
2. Double-click on the **Allocate** icon to start the ALC utility and open the Allocation panel (see Figure 4.8).
3. The ALC utility automatically displays the first OPEN-x FMT volume (in alphanumeric order) in the **VOLSER** field. If this is not the desired volume, select the desired volume from the drop-down list of volsers. If ALC could not find any OPEN-x FMT volumes, ALC displays the **HMDE format disk not found** message (see Figure 4.9).
4. Enter the name of the dataset being allocated in the **Dataset** field (maximum forty-four characters: A-Z, 0-9, @, #, ., \). Do not use any spaces or symbols other than @, #, ., and \.
5. Enter or select the size of the new dataset (number of cylinders, number of tracks) in the **Cylinder** and **Track** fields. The file size will be (# of cyl) + (# of tracks). The **Max.** button enters the maximum size for the new dataset in the **Cylinder** and **Track** fields based on the available capacity. The **Available Capacity** box displays the free space on the specified volume, so that you can select the appropriate size for the new dataset.
6. Enter or select the record format in the **Record format** field: F, FB, V, or VB.
7. Enter or select the record length in the **Record length** field:
   When record format = F, record length = block length.
   When record format = FB, record length = block length ÷ N (N = integer).
   When record format = V or VB, 5 ≤ record length ≤ (block length – 4).
8. Enter or select the block length in the **Block length** field. If block length = record length, select the **Copy** button to copy the record length into the **Block length** field.
   When record format = F or FB, 1 ≤ block length ≤ 32760.
   When record format = V or VB, 9 ≤ block length ≤ 32760
9. When all parameters for the new dataset are correct, select the **Start** button.
10. When the allocate operation completes successfully, the **Allocation complete** message is displayed (see Figure 4.10). If the allocate operation could not be started due to an error condition, the **Allocate check error** message is displayed. If the allocate operation did not complete successfully, one of the following error messages is displayed (see Figure 4.11) (n = system error code):

Open error! (n)	Open process error on the HMDEoto volume.
Seek error! (n)	Seek process error on the HMDEoto volume.
Read error! (n)	Read process error on the HMDEoto volume.
Write error! (n)	Write process error on the HMDEoto volume.
Close error! (n)	Close process error on the HMDEoto volume.

11. When you are finished allocating datasets on HMDEoto volumes, select **Close** to close the Allocation panel and exit the ALC utility.

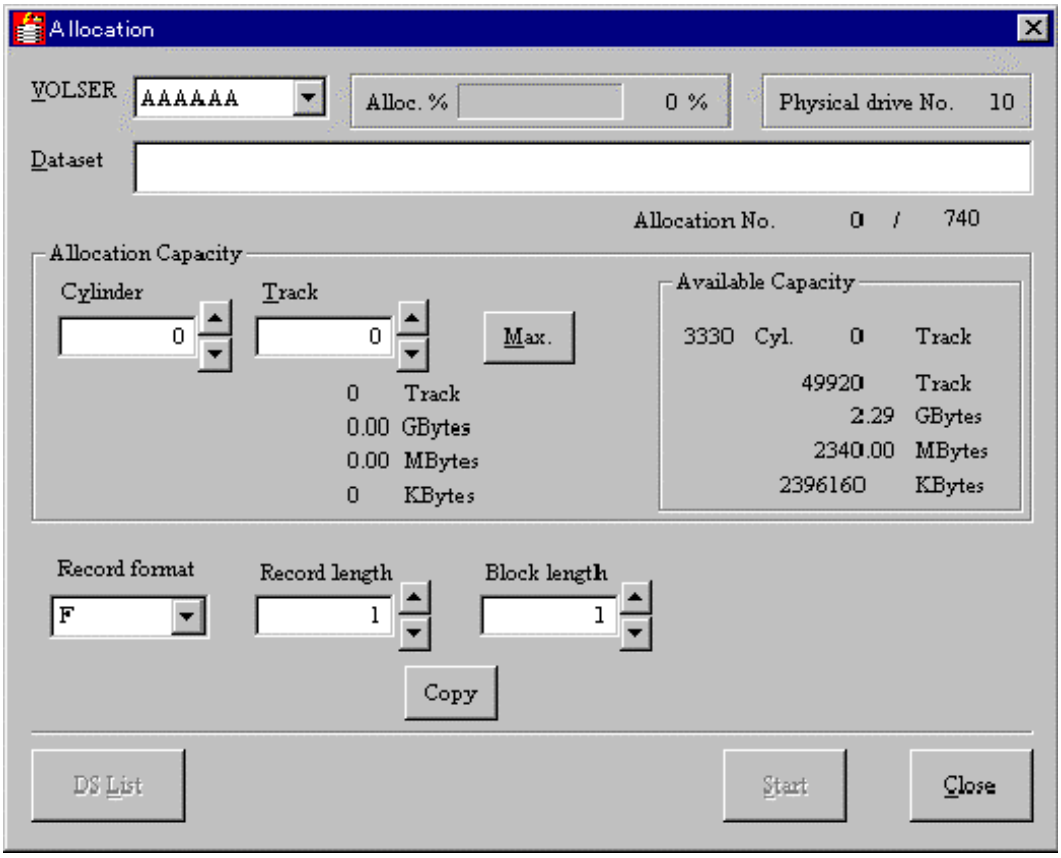


Figure 4.8 ALC Utility for Windows NT

Figure 4.9 ALC Disk Not Found Message

Figure 4.10 ALC Allocation Complete Message

Figure 4.11 ALC Error Message

4.4 Creating FCU Parameter Definition Files

The FCU parameter definition files are used to store FCU initiation parameter sets for pre-defined HMDE operations. If you plan to perform specific sets of HMDE operations more than once, you can create an FCU parameter definition file for each set of HMDE operations. The FCU program allows you to specify which FCU parameter definition file to load and execute, and you can also choose whether to execute all HMDE operations defined in the file or confirm each operation before starting it. You can create and edit the FCU parameter definition files interactively using the FCU GUI or manually using a text editor.

Table 4.3 lists the requirements for the FCU parameter definition files. Each set of FCU initiation parameters specifies the direction, source and target files, and FCU options (e.g., padding, delimiters) for a specific HMDE operation.

Table 4.3 FCU Parameter Definition File Requirements

Item	Requirement(s)
Default file name/location	For UNIX systems: **fcudata.param** in the directory containing the FCU program. For Windows NT: **fcudata.prm** in the directory containing the FCU program. ***Note:*** When upgrading from version 01-01-24 or earlier to version 01-01-36 or later on Windows NT, rename the file to change **.param** to **.prm**. To access the default FCU parameter definition file, leave the **param** option blank when starting FCU. To access a different parameter definition file, specify the file name (with complete path if necessary) when starting FCU. For Windows NT, the FCU parameter definition file must have the **.prm** file extension.
Maximum number of parameter sets	For UNIX systems: 999. For Windows NT: determined only by system memory.
FCU initiation parameters	Must be separated by at least one space character. Case sensitive (e.g., use **Yes**, not **YES** or **yes**, for the **Emp** and **RDW** parameters). Parameters (1)-(6) must be specified in order. Parameters (7)-(9) (optional) can be specified in any order (not before 1-6).
Comment lines (start with **#**) When using the FCU GUI	For UNIX systems: cannot be created using GUI. loaded but not displayed by GUI ("Parameter file:Comment line" displayed). can be changed to a normal parameter line using GUI. included in number of parameter sets (each comment counts as one set). For Windows NT: skipped by FCU (not processed). cannot be created, edited, or deleted using GUI (must use text editor). cannot be displayed by GUI ("Parameter file:Comment line" is displayed). not included in line count.
Comment lines (start with **#**) When not using the FCU GUI	For UNIX only: skipped by FCU (not processed). not included in number of parameter sets (max = 999).
Space lines	Not allowed with FCU GUI (remove space lines if upgrading from 01-01-24 or earlier to 01-01-36 or later on Windows NT system). Allowed and skipped when using FCU without the GUI (using the **-nw** option).

Figure 4.12 illustrates the structure of an FCU parameter definition file. Table 4.4 lists the requirements for the FCU initiation parameters. To define HMDEoto operations, first define the HMDEotm operations which transfer the data from the source files to the intermediate datasets, and then define the HMDEmto operations which transfer the data from the intermediate datasets to the target files. Do not define HMDEoto operations which access OPEN-3 FMT volumes and HMDE operations which access -A, -B, or -C HMDE volumes in the same FCU parameter definition file. Since FCU can only access one HMDE volume definition file at a time, each FCU parameter definition file must contain either HMDE operations which access OPEN-3 FMT volumes or HMDE operations which access -A, -B, or -C volumes, but not both.

```
mto   VSN:dataset name        Open-system file name   CC  PAD  DEL  Emp=Yes  RDW=Yes  VSE=RF,RL,BL
mto   VSN:dataset name        Open-system file name   CC  PAD  DEL  Emp=Yes  RDW=Yes  VSE=RF,RL,BL
otm   Open-system file name   VSN:dataset name        CC  PAD  DEL  Emp=Yes           VSE=RF,RL,BL
otm   Open-system file name   VSN:dataset name        CC  PAD  DEL  Emp=Yes           VSE=RF,RL,BL
 (1)       (2)                     (3)               (4) (5)  (6)  (7)       (8)      (9)          (10)
 :
end
(11)
```

Figure 4.12 FCU Parameter Definition File

Table 4.4 FCU Initiation Parameter Requirements (continues on next page)

Number	Name	Function	Options
(1)	HMDE direction	Required. Specifies the file transfer direction.	**mto** for HMDE mainframe-to-open. **otm** for HMDE open-to-mainframe.
(2)	Source file	Required. Specifies the source file.	**VSN:dataset** for mainframe source dataset: VSN = six-character volume serial number colon = separates VSN and dataset name dataset = dataset name (44 chars max, no spaces) **file name** for open-system source file: Space characters not allowed. Specify path (absolute or relative) if not in current directory: UNIX path: /directory_name/…/file_name Windows NT path: drive:\directory_name\…\file_name
(3)	Target file	Required. Specifies the target file.	Same as parameter (2): **VSN:dataset** or **file_name**.
(4)	Code conversion	Required. Specifies code conversion.	**EA** to use default EBCDIC-ASCII code conversion table for mto or otm. **EcA** to use default EBCDIC-ASCII code conversion table for oto. **No** for no code conversion. **File_name** to use your own code conversion table (see section 3.1.1).
(5)	Padding	Required. Specifies padding option.	**Yes** for HMDEmto with padding. **No** for HMDEmto without padding and for HMDEotm.

Table 4.4 FCU Initiation Parameters (continued)

Number	Name	Function	Options
(6)	Delimiter	Required. Specifies delimiter option and type.	For HMDEmto operations: UNIX: **CR** Carriage return (CR) is added as a delimiter. **LF** Line feed (LF) is added as a delimiter. **No** No delimiter is added. Windows NT: **CRLF** = CR + LF is added as a delimiter. For HMDEotm operations: UNIX: **CR** Data up to CR is cut off as data entity. **LF** Data up to LF is cut off as data entity. **No** Data is cut off according to dataset record length. Windows NT: **CRLF** Data up to CR + LF is cut off as data entity. **No** Data is cut off according to dataset record length.
(7)	Empty file	Optional. Enables processing of empty source files.	This parameter is optional. If not specified, **Emp=No** is assumed. **Emp=Yes** Execute the data transfer even if the source file is empty. HMDEmto target file size = 0. HMDEotm target dataset will contain only EOF. **Emp=No** Execute the data transfer. If the source file is empty, the HMDE operation will be rejected with an error.
(8)	Record description word	Optional. Specifies if FCU adds the record description word. HMDEmto only. ***WARNING:*** Data transferred to open using RDW=Yes and then transferred back to S/390 is not compatible with the original dataset.	This parameter is optional. If not specified, **RDW=No** is assumed. Do not specify this parameter for HMDEotm operations. **RDW=Yes** Add record description word to each record. CC, PAD, and DEL must be **No** (if not, error). Direction must be **mto** (if not, error). Source dataset must be variable length (if not, **RDW=Yes** is ignored). **RDW=No** Do not add record description word to each record. Outputs only the data entity for each record.
(9)	VSE record	Optional. Specifies the RF, RL, and BL for VSE datasets. HMDEmto and HMDEotm only.	This parameter is optional. If not specified, the VTOC must specify the RF, RL, and BL. Do not specify this parameter for HMDEoto using ALC-generated datasets on OPEN-3 FMT volumes. **VSE=RF,RL,BL** RF, RL, and BL must be separated by a comma (,) and no spaces. **RF** **F** fixed-length and unblocking **FB** fixed-length and blocking **V** variable-length and unblocking **VB** variable-length and blocking **RL** record length in bytes (decimal) When RF = F: RL = BL When RF = FB: RL = [BL/n] (n is an integer) When RF = V or VB: $5 \leq RL \leq [BL - 4]$ **BL** When RF = F or FB: 1 through 32760 When RF = V or VB: 9 through 32760
(10)	Carriage return	Required. Marks end of parameter set.	Press **Return** (**Enter** for Windows NT) at the end of each line.
(11)	End of file	Optional. Marks end of parameter file.	**end** This parameter is optional.

Chapter 5 Performing FCU File Transfer Operations

HMDE file transfer operations are performed using the FCU GUI software installed on the open-system host(s) attached to the 9900 subsystem. The FCU GUI enables you to perform file transfer operations interactively, provides access to detailed information on the HMDE source datasets and files, and displays error information for HMDE operations. The FCU GUI also allows you to create and modify FCU parameter definition files interactively.

When you perform HMDE operations which access datasets on -A, -B, or -C HMDE volumes, FCU must have access to the HMDE volume definition file which defines these volumes. When you perform HMDE operations which access ALC-generated datasets on OPEN-x FMT HMDE volumes, FCU must have access to the separate HMDEoto volume definition file which defines the OPEN-x FMT volumes (see section 4.1). Since FCU can only access one HMDE volume definition file at a time, the FCU parameter definition files must also keep operations using OPEN-x FMT volumes separate from operations using -A, -B, or -C HMDE volumes. Before you start FCU GUI operations, make sure that the desired HMDE volume definition file is available (**datasetmount.dat** in current directory) and that the desired FCU parameter definition file contains HMDE operations which access the volumes defined in the HMDE volume definition file. FCU will not be able to perform operations which access volumes which are not defined in the current HMDE volume definition file.

The FCU GUI for UNIX-based platforms and the FCU GUI for Windows NT are significantly different. Section 0 describes and provides instructions for using the FCU GUI for UNIX. Section 5.2 describes and provides instructions for using the FCU GUI for Windows NT. For information on using FCU from the UNIX command line (without the GUI), see Appendix B.

For information on using the FAL C functions (Visual C++ for Windows NT) which enable user programs on the open-system host to access S/390 datasets on HMDE volumes, see Chapter 6.

Note: FAL/FCU cannot handle files larger than 2 GB (an error will be returned).

5.1 Using the FCU GUI for UNIX

The FCU GUI enables you to perform HMDE file transfer operations interactively and provides access to detailed information on the datasets/files in the specified HMDE source volume/directory. The FCU GUI displays the HMDE operations in the FCU parameter definition file (if specified), allows you to modify the FCU parameter definition file interactively, and also allows you to enter FCU parameters and perform HMDE operations manually. The FCU GUI also displays the error information for HMDE operations.

5.1.1 Starting the FCU GUI for UNIX

To start the FCU GUI program for UNIX-based platforms:

1. At the UNIX command line prompt, enter: **fcu [-nc] [param]**

   The **-nc** option (**nc** = no checking) tells FCU to execute all specified HMDE operations without requesting confirmation for FCU parameters or checking for existing HMDEmto target files. If you want to bypass these confirmations, enter **-nc**.

   The **param** option tells FCU whether to use the FCU parameter definition file or a specific FCU initiation parameter set to perform HMDE operations. The **param** option must have one of the following three values:

   – [blank]. If you want to use the default FCU parameter definition file (**fcudata.param** in the current directory), leave the **param** option blank (do not enter anything).

   – **file_name**. If you want to use a different FCU parameter definition file, enter the file name with complete path (absolute or relative) if not in the current directory.

   – **-P + parameters**. If you want to perform one specific HMDE operation, enter **-P** followed by the FCU initiation parameter set (e.g., **mto VSN:dataset targetfile No No No**) for the desired HMDE operation. The **-P** option requires the **-nc** option.

   For example:

   – If you want to use the default FCU parameter definition file and check the parameters and HMDEmto target files, enter: **fcu**

   – If you want to use the default FCU parameter definition file and perform all operations without checking parameters or HMDEmto target files, enter: **fcu -nc**

   – If you want to use a different FCU parameter definition file and perform all operations without checking parameters or HMDEmto target files, enter: **fcu -nc filename**

   – If you want to perform one specific HMDE operation, enter: **fcu -nc -P [parameters]**

   Note: The following warnings may appear during FCU startup. These warnings do not affect FCU and can be ignored.

   Warning: Missing characters in String to FontSet conversion.
   Warning: Cannot convert string "-dt-interface system-medium-r-normal -m*-*-*-*-*-*-*-*-*" to type FontSet.

2. The FCU GUI program now starts loading. The FCU version and copyright screen (see Figure 5.1) is displayed while FCU is loading. When FCU is finished loading, the FCU main panel (see section 5.2.2) is displayed.

3. If you specified the **-nc** option, FCU processes all specified operations, overwrites existing mto target files, terminates, and displays any error information at the UNIX prompt.

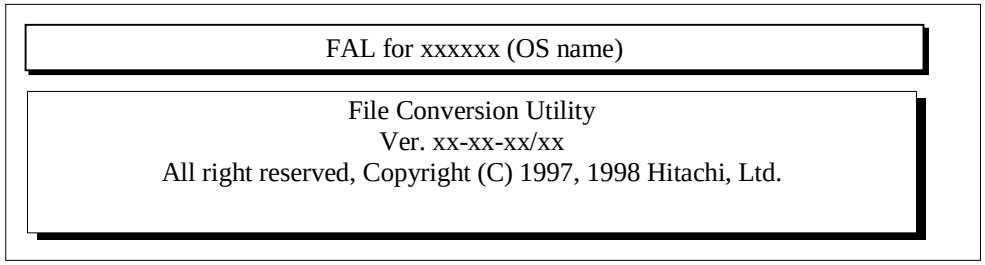


Figure 5.1 FCU Version and Copyright Screen (UNIX)

5.1.2 FCU Main Panel for UNIX

The FCU main panel (see Figure 5.2) opens when the FCU GUI program is finished loading. The FCU main panel displays the FCU initiation parameter sets in the specified FCU parameter definition file (if available), allows you to perform HMDE operations, and provides access to all FCU functions.

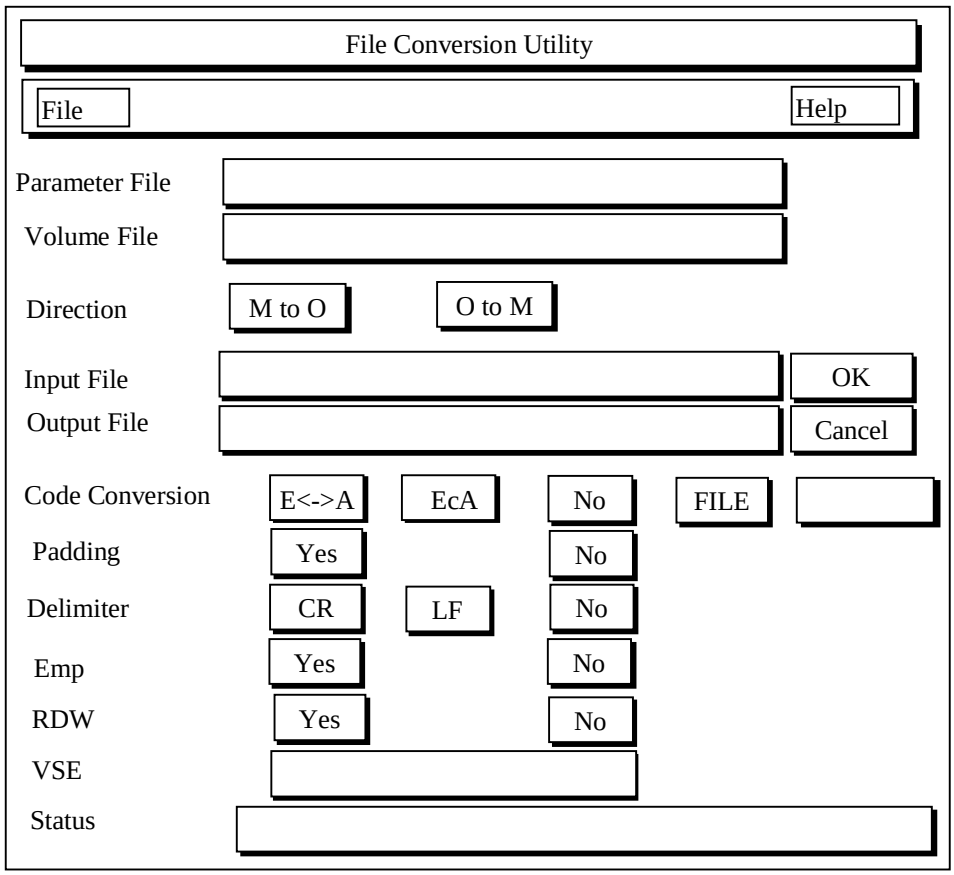


Figure 5.2 FCU Main Panel for UNIX platforms

The **File** and **Help** buttons display the File menu commands and Help menu commands. These commands are described later in this section.

The **Parameter File** field displays the FCU parameter definition file (see section 4.4) that you specified by the **param** option when you started FCU. If this field is blank, FCU could not find the default or specified FCU parameter definition file. If you want to use an FCU parameter definition file, you can enter the desired file name in this field (complete path if not in the current directory). If you do not want to use an FCU parameter definition file, you can leave the **Parameter File** field blank and enter the FCU initiation parameters manually.

The **Volume File** field displays the HMDE volume definition file (see section 4.1). This file must have the default name and location (**datasetmount.dat** in current directory). If this field is blank, FCU could not find the file and will not be able to perform HMDE operations. In this case, exit FCU, and create the HMDE volume definition file as described in section 4.1.

When FCU starts up, the first set of FCU initiation parameters is automatically loaded from the specified FCU parameter definition file (unless the file is not found). If desired, you can change any of the parameters, or you can use the **File-Load** command to load the next parameter set. The FCU initiation parameters are:

- **Direction**. The **Direction** buttons allow you to select the desired direction for the HMDE operation: **M to O** = HMDEmto, **O to M** = HMDEotm.
- **Input File**. The **Input File** field allows you to enter the name of the HMDE source file. For HMDEmto, enter the S/390 volser and dataset name (**VSN:dataset**). For HMDEotm, enter the UNIX file name (with complete path if not in the current directory).
- **Output File**. The **Output File** field allows you to enter the name of the HMDE target file. For HMDEmto, enter the UNIX file name (with complete path if not in the current directory). For HMDEotm, enter the VSN and dataset name (**volser:dataset**).
- **Code Conversion**. The **Code Conversion** buttons allow you to select the desired code conversion option (see section 3.1.1): **E<->A** = default code conversion table, **EcA** = default code conversion table (for HMDEoto only), **No** = no code conversion, **File** = enter the file name of your conversion table (with complete path if not in current directory).
- **Padding**. The **Padding** buttons allow you to select the desired padding option (see section 3.1.3): **Yes** = padding, **No** = no padding.
- **Delimiter**. The **Delimiter** buttons allow you to select the desired delimiter option (see section 3.1.4): **CR** = carriage return, **LF =** line feed, **No =** no delimiters.
- **Emp**. The **Emp** buttons allow you to select the empty file option (see section 3.1.5): **Yes** = source file is empty, **No** = source file is not empty.
- **RDW**. The **RDW** buttons allow you to select the record description word option (mto only) (see section 3.1.5): **Yes** = add RDW to each record (**Code Conversion**, **Padding**, and **Delimiter** must be **No**), **No** = do not add RDW to each record.
- **VSE**. The **VSE** field allows you to enter the VSE record information: **RF,RL,BL** Use a comma (no spaces) between each value. See section 3.1.6 for further information on the VSE record option values.

The **OK** button starts the specified HMDE operation. The **Cancel** button removes the values entered by the user and returns the FCU main panel to the initial settings. (The **Cancel** button does not cancel the HMDE operation in progress.) Be careful not to click **OK** or **Cancel** more than once. The **Status** field displays the status of the requested HMDE operation:

- **Now checking** = FCU is executing a dataset search or file attribute check. If you specified the **-nc** option when you started FCU, this check does not occur.
- **Overwrite ? (OK/Cancel)** is displayed if the HMDEmto target file already exists. Click **OK** to overwrite the existing file, or click **Cancel** to cancel the requested operation. If you specified the **-nc** option when you started FCU, this confirmation does not occur.
- **x%** = The requested HMDE operation is x% complete.
- **Complete** = The requested HMDE operation completed successfully.
- **Error**. The **Status** field also displays error information for FCU and HMDE operations. See Appendix C for further information on error conditions.

5.1.3 File Menu Commands

The **File** menu provides access to the following FCU functions:

- **Load**. This command loads the parameter sets from the specified FCU parameter definition file onto the FCU main panel. Each time you select **Load**, the next set of parameters is loaded. If you enter a file name in the **Parameter File** field, the **Load** command opens the file and loads the first parameter set (or creates the new file). If the FCU parameter definition file is empty or was not found, FCU ignores this command.
- **Save**. This command saves the FCU parameter definition file. If no FCU parameter set was previously loaded, the current parameter set is added to the file. If a parameter set was previously loaded and you made changes, the current parameter set overwrites and replaces the previously loaded parameter set. If you make changes and do not select **Save**, FCU will discard your changes when you select **Load** or **Exit**.
- **Delete**. This command deletes the currently loaded parameter set from the FCU parameter definition file. If the FCU parameter definition file does not yet exist or does not contain the parameter set on screen, FCU ignores this command.
- **Exit**. This command closes the current FCU parameter definition file (unsaved changes are discarded), and then closes the FCU program.

5.1.4 Help Menu Commands

The **Help** menu provides access to the following FCU functions:

- **Volume**. This command displays the contents of the HMDE volume definition file (see Figure 5.3), so that you can verify that the HMDE volumes are properly defined.
- **MF-File**. This command displays the dataset information for each dataset in the specified mainframe (MF) volume (see Figure 5.4). The VSN must be entered in the **Input File** field (for HMDEmto) or **Output File** field (for HMDEotm) on the FCU main panel.
  - Dataset name: An asterisk (*) before the dataset name indicates that FAL/FCU can process the dataset. A dash (-) indicates that FAL/FCU cannot process the dataset. A question mark (?) indicates that FCU can process the dataset only if the VSE record option is used to specify the RF, RL, and BL.
  - Dataset organization (DO) type: SAM, DAM, PAM, VSAM, ??? = unknown.
  - Record format (RF): F = fixed length, V = variable length, U = undefined length, S = spanned record, ? = unknown.
  - Block length (BL): in bytes
  - Record length (RL): in bytes
  - Dataset size (DS): in tracks
- **UX-File**. This command displays the UNIX (UX) files in the directory specified in the **Input** or **Output File** field on the FCU main panel (see Figure 5.5). If no directory is specified in the **Input File** or **Output File** field, FCU displays the files in the current directory. If a nonexistent directory is specified, FCU will return an error.
- **Error**. This command opens the error information panel, which displays the FAL, FCU, and system error codes/messages (see Figure 5.6).
- **OnVersion**. This command displays the FCU version and copyright information screen.

```
/dev/rdsk/cxtydz  volser  3390-3B
/dev/rdsk/cxtydz  volser  3390-3A
/dev/rdsk/cxtydz  volser  3390-3C
/dev/rdsk/cxtydz  volser  3380-KB
/dev/rdsk/cxtydz  volser  3380-KA
/dev/rdsk/cxtydz  volser  3380-KC
 :
 :
end
```

Figure 5.3 Help-Volume Display (HP-UX shown)

```
Dataset Information : VSN = xxxxxx   Device Emulation Type = 3390-3B

 Dataset Name           DO      RF    BL      RL      DS

*SAMFILE01.FIX          SAM     F     4096    128     150       ← Can be processed by FCU.
-DAMFILE.F              DAM     F     4096    128     30        ← Cannot be processed by FCU.
*SAMFILE02.VAR          SAM     V     4000    80      50        ← Can be processed by FCU.
-PAMFILE                PAM     F     5000    100     200       ← Cannot be processed by FCU.
-VIRTUALSTORAGEACCESS   VSAM    V     32768   4096    3000      ← Cannot be processed by FCU.
-UNDEFSAMFILE           SAM     U     8000    200     80        ← Cannot be processed by FCU.
-SAMFILESPANNED         SAM     S     8192    8192    300       ← Cannot be processed by FCU.
```

Figure 5.4 Help MF-File Display

```
            UNIX FILE LIST : DIR = /aaaaa/bbbbb/cccc

ddddd.dd        eeeeee        ffffff.fffffff
hhhhh.hhhh      zzzzzz.z      xxxx.x
yyyyyyyyy
```

Figure 5.5 Help UX-File Display

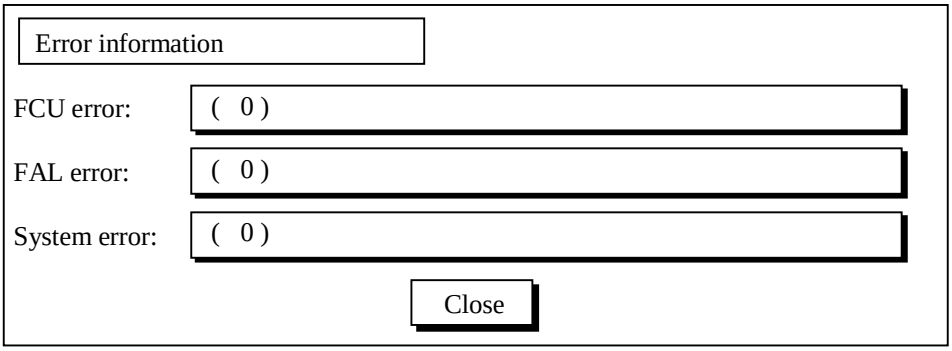


Figure 5.6 Error Information Display

5.1.5 Creating an FCU Parameter Definition File Using FCU for UNIX

To create an FCU parameter definition file using the FCU GUI for UNIX:

1. Start the FCU GUI for UNIX by entering **fcu** (see section 5.1.1). Do not specify the **-nw**, **-nc**, or **param** option.
2. When the FCU main panel opens (see section 5.1.2), enter the desired file name in the **Parameter File** field (with complete path if you do not want to save the file in the current directory).
3. If you plan to perform HMDE operations while you are creating the FCU parameter definition file, make sure that the **Volume File** field displays the correct HMDE volume definition file (**datasetmount.dat**). If not (or if incorrect), FCU will not be able to perform HMDE operations, but you can still create a new FCU parameter definition file.
4. Select the **File-Load** command to open the new file.
5. Enter the desired FCU initiation parameters for the first HMDE operation:
   - Select the file transfer direction using the **M to O** button or **O to M** button.
   - Enter the source and target files in the **Input File** and **Output File** fields (**VSN:dataset**, **filename** with complete path if not in current directory).
   - Select the desired FCU file transfer options: **Code Conversion**, **Padding**, **Delimiter**, **Emp**, **RDW**, and **VSE**. See section 3.1.1 for further information on these options.
6. When the FCU initiation parameters are correct, select the **File-Save** command to add this parameter set as the first line in the new FCU parameter definition file. If the HMDE volume definition file is correct, you can perform the operation now by clicking **OK**. If the **OK** button is not enabled, the parameter set has not been saved in the file.
7. Select the **File-Load** command to load the next line. The **Status** field should indicate that you are at the end of the file. The FCU GUI for UNIX only allows you to add new lines when you are at the end of the file (right after the last line).
8. Repeat steps (5), (6), and (7) to add each parameter set to the new FCU parameter definition file. Make sure to keep HMDE operations which use OPEN-x FMT volumes in a separate FCU parameter definition file from operations which use -A, -B, -C volumes.
9. If you need to modify an existing line, go to the line to be modified using the **File-Load** command, change the parameters as needed, and then use the **File-Save** command to replace the line that was loaded.
10. If you need to insert a new line between existing lines, use a text editor later to edit the file. You cannot add a new line between existing lines using the FCU GUI for UNIX.
11. When you are finished adding lines to your new FCU parameter definition file, make sure that you have selected the **File-Save** command for the last parameter set you added or modified, and then select the **File-Exit** menu command to close the file and exit FCU.

5.1.6 Performing HMDE Operations Using the FCU GUI for UNIX

To perform HMDE file transfer operations using the FCU GUI for UNIX:

1. If you will be performing HMDEmto operations:
   a) Make sure that the source datasets are located on the desired HMDE volume(s). If you will not be using an existing FCU parameter definition file, write down the VSN:dataset of the source dataset and the complete path and file name of the target file for each HMDEmto operation.
   b) Verify that the HMDEmto target files do not already exist (or can be overwritten).
   c) Vary the HMDEmto volume(s) and channel path(s) offline from the S/390 host.
2. If you will be performing HMDEotm operations:
   a) Make sure that the source files are located on the desired HMDE volume(s). If you will not be using an existing FCU parameter definition file, write down the complete path and file name of the source file and the VSN:dataset of the target dataset for each HMDEotm operation.
   b) Create and allocate the target datasets. This ensures that the target dataset is registered in the VTOC. Make sure to allocate enough space and to use the appropriate record format and record length for the data to be transferred.
   c) Vary the HMDEotm volume(s) and channel path(s) offline from the S/390 host.
3. If you will be performing HMDEoto operations:
   a) If you will not be using an existing FCU parameter def. file, write down the complete path and file name of the source and target files for each HMDEotm/mto operation.
   b) Allocate the intermediate datasets on the HMDEoto volume(s). Use the ALC utility on OPEN-3 FMT volumes (see section 4.3). Make sure to allocate enough space and to use the appropriate record format and record length for the data to be transferred.
   c) Verify that the HMDEoto target files do not already exist (or can be overwritten).
4. Make sure that the desired HMDE volume definition file (HMDEoto only, or HMDEmto and HMDEotm) is available for use by FCU (**datasetmount.dat** in current directory).
5. Start FCU with the desired options (see section5.1.1). ***Note:*** If you specify the **-nc** option, FCU performs all specified operations continuously, then self-terminates and displays any error information at the UNIX prompt.
6. When the FCU main panel opens, make sure that the **Volume File** field displays the HMDE volume definition file (**datasetmount.dat**). If the HMDE volume definition file is not displayed (or if incorrect), FCU will not be able to perform HMDE operations.
7. Make sure that the **Parameter File** field displays the desired FCU parameter definition file. If not, enter the desired FCU parameter definition file name (with complete path if not in the current directory), and select the **File-Load** command to open the file. If you want to create a new file using the FCU GUI, see section 5.1.3 for instructions.

8. The FCU main panel displays the first/next parameter set in the specified FCU parameter definition file. If you want to perform this HMDE operation, click **OK**. If not:
   a) You can load the next parameter set using the **File-Load** command.
   b) You can delete the current parameter set from the FCU parameter definition file using the **File-Delete** command. The next parameter set loads automatically.
   c) You can modify the current parameter set as follows: change the FCU parameters as needed, and then use the **File-Save** command to save your changes in the FCU parameter definition file (replaces the previously loaded parameter set).
   d) You can add a new parameter set to the end of the file as follows: select **File-Load** until you reach the end of the file, enter the desired parameters, and then select **File-Save** to add the new line at the end. If you want to insert a new line between existing lines, edit the FCU parameter definition file later using a text editor.
9. When the desired HMDE operation is displayed, click **OK** to start the operation. (If the **OK** button is not enabled, you have not saved the current parameter set.)
10. If you started an HMDEmto operation and the target file already exists, FCU requests overwrite confirmation. Click **OK** to overwrite the target file, or click **Cancel** to cancel the operation.
11. When FCU starts the operation, the **Status** field displays the progress of the operation. If desired, while the operation is in progress, you can load another parameter set and click **OK** to start the next operation right after the current operation completes. ***Note:*** Be careful when doing this. If you click buttons or menu commands while an operation is in progress, FCU will save and execute those commands when the current operation completes.
12. When the HMDE operation is complete, the **Status** field displays **Complete**. If an error occurred, the error information display (refer toFigure 5.6) opens automatically to display the error. See Appendix C for further information on error conditions.
13. FCU does not load the next operation automatically. To perform another HMDE operation, select **File-Load**, and repeat steps (8) through (12). To exit FCU, select the **File-Exit** command.

5.2 Using the FCU GUI for Windows NT

5.2.1 Starting the FCU GUI for Windows NT

To start the FCU GUI program for Windows NT:

1. Log on with Administrator access privileges.
2. Start the FCU GUI as follows: click **Start-Programs-FCU-FCU**, or open the **c:** folder and double-click on **FCU**, or create a shortcut for **FCU** on the desktop. ***Note:*** Do not start FCU by dragging and dropping an FCU parameter definition file on the FCU program icon. FCU program operation cannot be guaranteed.
3. If you want to specify any of the FCU options, start FCU from the command line (DOS prompt) as follows: go to the FCU directory (containing **fcu.exe** and **datasetmount.dat**), and enter **fcu [-nc] [-cl] [param]**

   The **-nc** option is the same as for UNIX: all specified HMDE operations are performed without confirmation of FCU parameters or HMDEmto target file overwrite.

   The **-cl** option specifies that all FCU log files will be cleared before starting.

   The **param** option is the same as for UNIX:

   If you want to open a new untitled FCU parameter definition file when you start FCU, leave the **param** option blank.

   If you want to load an FCU parameter definition file when you start FCU, enter the file name with complete path if the file is not in the current directory.
4. The FCU GUI program now starts loading. The FCU version and copyright screen (seeFigure 5.7) is displayed while FCU is loading. When FCU is finished loading, the FCU main panel is displayed (see section 5.1.2).
5. If you started FCU from the DOS prompt and specified the **-nc** option, FCU processes all specified operations, overwrites existing HMDEmto target files, and then terminates and displays any error information at the DOS prompt.

Figure 5.7 FCU Version and Copyright Screen (Windows NT)

5.2.2 FCU Main Panel for Windows NT

The FCU main panel (see Figure 5.8) opens when FCU is finished loading. The FCU main panel displays the FCU parameter definition file (or **Untitled** if no file was specified), allows you to perform HMDE operations, and provides access to all FCU functions.

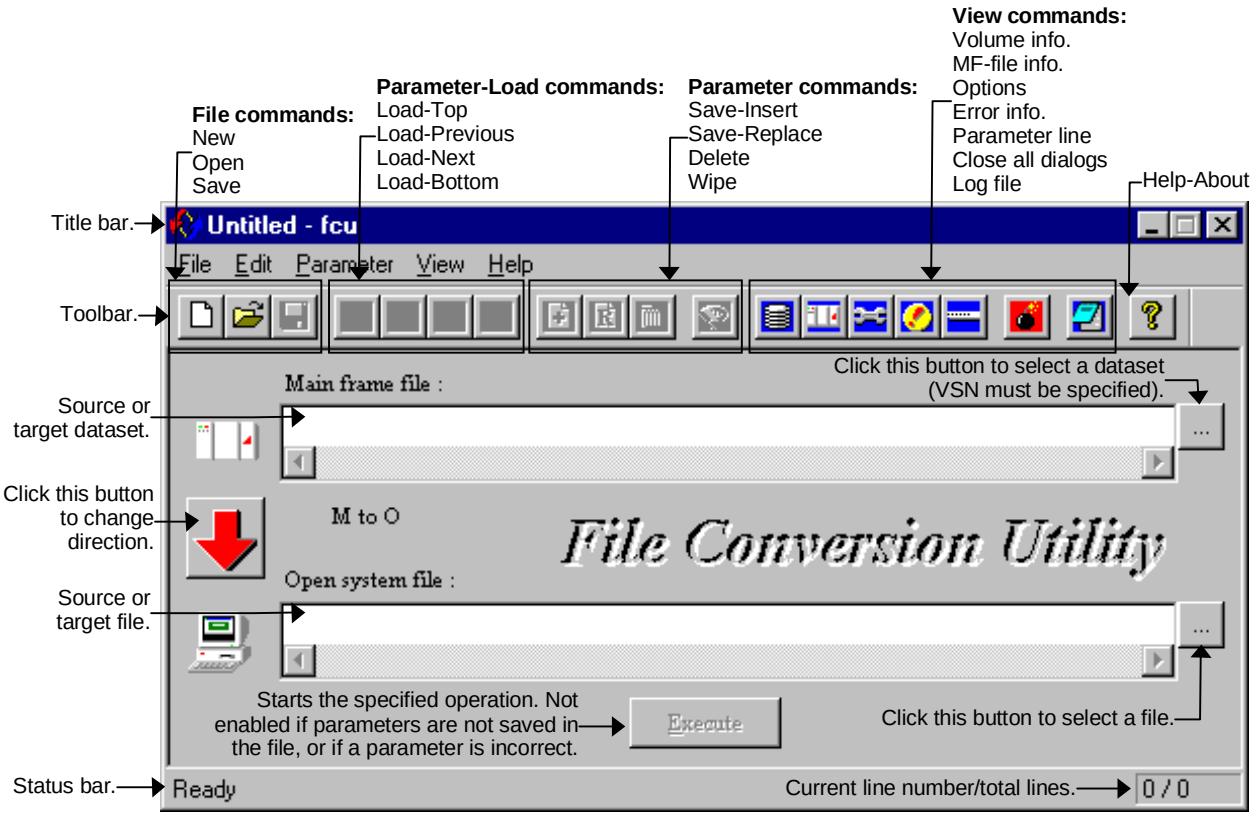


Figure 5.8 FCU Main Panel for Windows NT

The FCU title bar displays the current FCU parameter definition file. The toolbar provides speed buttons for the commonly used FCU functions. The status bar displays the current line number and total number of lines in the current FCU parameter definition file. The **Main frame file** and **Open-system file** fields display the files to be transferred (no spaces allowed). The file selection buttons ( ) allow you to select the desired MF dataset and NT file.

The **File** menu provides access to the following FCU functions:

- The **File-New** command (Ctrl+N) opens a new FCU parameter def. file (untitled.prm).
- The **File-Open** command (Ctrl+O) opens an existing FCU param. def. file (filename.prm).
- The **File-Save** command (Ctrl+S) saves the current FCU parameter definition file. Deleted and replaced lines are discarded, inserted lines are added, and all lines after **end** are discarded. ***Note:*** This command does not save the current parameter set.
- The **File-Save As…** command saves the current FCU parameter definition file with a different file name and/or location.
- The **File-Exit** command (Ctrl+X) exits the FCU software.

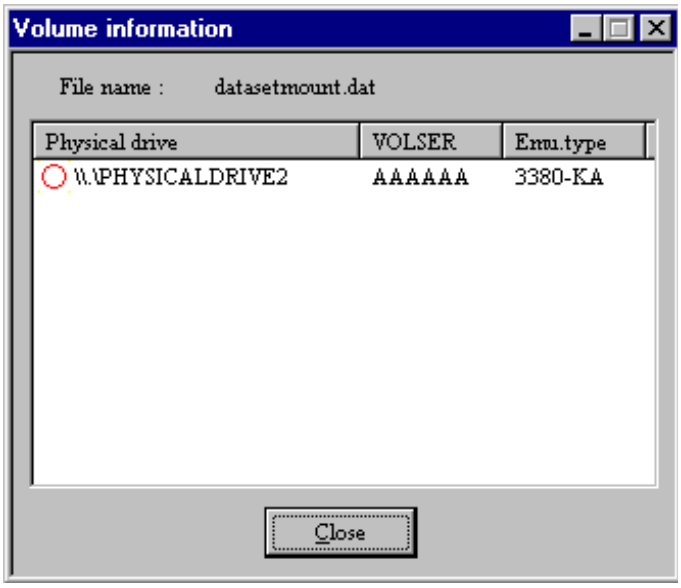
The **Edit** menu is reserved for future enhancement and is not yet enabled.

The **Parameter** menu provides access to the following FCU functions:

- The **Parameter-Load** command loads the **Previous**, **Next**, **Top**, and **Bottom** parameter lines from the current FCU parameter definition file. The FCU main panel status bar updates the current line number when any **Parameter-Load** command is executed.
- The **Parameter-Save** command allows you to either **Insert** the parameter set being displayed into the current FCU parameter definition file, or **Replace** the current parameter set (previously loaded) with the parameter set being displayed. If you do not select this command, your parameter changes will not be saved. ***Note:*** This command does not save the current FCU parameter definition file (you must use **File-Save/Save As**).
- The **Parameter-Delete** command deletes the current parameter set from the current FCU parameter definition file. The line is not permanently deleted until you save the current FCU parameter definition file using the **File-Save** command.
- The **Parameter-Wipe** command clears all FCU initiation parameters displayed on screen, so that you can input new parameters easily. This command does not delete the current parameter set.

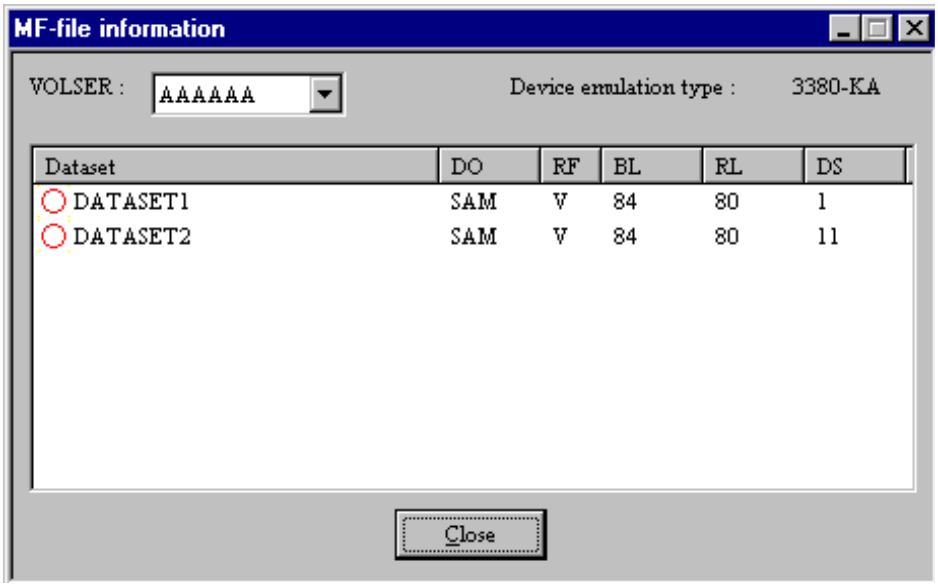
The **View** menu provides access to the following FCU functions:

- The **View-Toolbar** and **View-Status bar** commands display/hide the FCU toolbar and status bar. The toolbar provides speed buttons for the commonly used FCU functions. The status bar displays the current line number and total number of lines in the current FCU parameter definition file.
- The **View-Volume information…** command opens the FCU Volume Information panel (see Figure 5.9), which displays the contents of the HMDE volume definition file. A ○ displayed next to a volume indicates that the volume definition is correct and FCU can access the volume. An ✕ displayed next to a volume indicates that the volume definition is not correct and FCU cannot access the volume.
- The **View-MF-file information…** command displays the following information for the mainframe (MF) files (datasets) in the volume specified in the **Mainframe file** field on the FCU main panel (see Figure 5.10):
  - ○ = the dataset can be processed by FCU.
    ✕ = the dataset cannot be processed by FCU.
    ? = the dataset can be processed by FCU only if the VSE record option is used to specify the RF, RL, and BL.
  - **Dataset** = dataset name
  - **DO** = dataset organization type: **SAM**, **DAM**, **PAM**, **VSAM**, **?** (other than above)
  - **RF** = record format: **F** (fixed-length), **V** (variable-length), **U** (undefined), **S** (spanned), **?** (other than above)
  - **BL** = block length
  - **RL** = record length
  - **DS** = dataset size (in number of tracks)



Note: This example shows an HMDE volume definition file which defines only one HMDE volume.

Figure 5.9 Volume Information Panel



Note: This example shows an HMDE volume (3380-KA, VSN = AAAAAA) which contains only two datasets.

Note: When this panel is opened using the **Mainframe file** selection button ( ), only SAM datasets are displayed.

Figure 5.10 MF-File Information Panel

- The **View-Option…** command opens the Option panel (see Figure 5.11), which allows you to view/change the FCU file transfer options (code conversion, padding, delimiter, empty file, RDW, and VSE), continuous execution option, and clear log file option.
  - Select the **EBCDIC<-->ASCII** button to use the default code conversion table (**EA**, **EcA**). Select **No** for no code conversion (**No**). Select **External table file** and enter the file name with path if not in the current directory (e.g., **/directory/filename.tbl**).
  - Check the **Padding** box to request the padding option (**Yes**).
  - Check the **Delimiter** box to request the delimiter option (**Yes**).
  - Check the **Empty** box to request the empty file option (**Emp=Yes**).
  - Check the **Record Description Word** box to request the RDW option (**RDW=Yes**). ***Note:*** If the **Record description word** box is checked, the **EBCDIC<-->ASCII** button and the **Padding** and **Delimiter** boxes are ignored.
  - Check the **VSE** box and enter the RF, RL, and BL to request the VSE record option.
  - Check the **Continuous execution** box to tell FCU to process the rest of the FCU initiation parameter sets in the specified FCU parameter definition file without stopping after each completed operation (equivalent to the **-nc** option starting at the desired line). FCU will execute all lines from the current line to the **end**. If you do not check the **Continuous execution** box, FCU will stop after each operation.
  - Check the **Clear log file before execution** box to clear the log file for the current FCU parameter definition file (e.g., fcudata.prm.log). The user should clear the FCU log files as needed to decrease the file size and save space on the current drive.

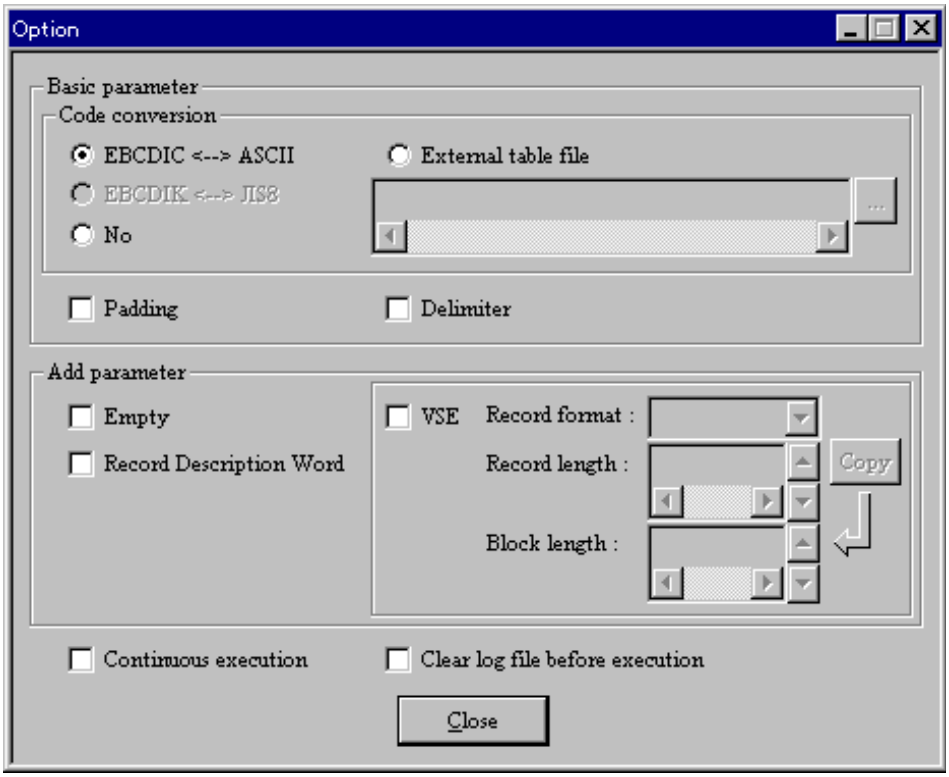


Figure 5.11 Option Panel

- The **View-Error information…** command opens the Error Information panel (see Figure 5.12), which displays the most recent error information (error code and message) for FCU, FAL, and system errors.
- The **View-Parameter line…** command opens the Parameter Line panel (see Figure 5.13), which displays the current line (parameter set) in the current FCU parameter definition file.
- The **View-Close all dialogs** command closes all open panels except the FCU main panel.
- The **View-Log file** command opens the log file for the current FCU parameter definition file using the Windows Notepad text editor (see Figure 5.14). The log file contains the parameter sets executed, the date and time of execution, the result of each operation, and the error information (FCU, FAL, and Sys error codes) for each operation.

The **Help-About FCU…** menu command opens the FCU version and copyright screen (refer to), so that you can check the FCU program version information.

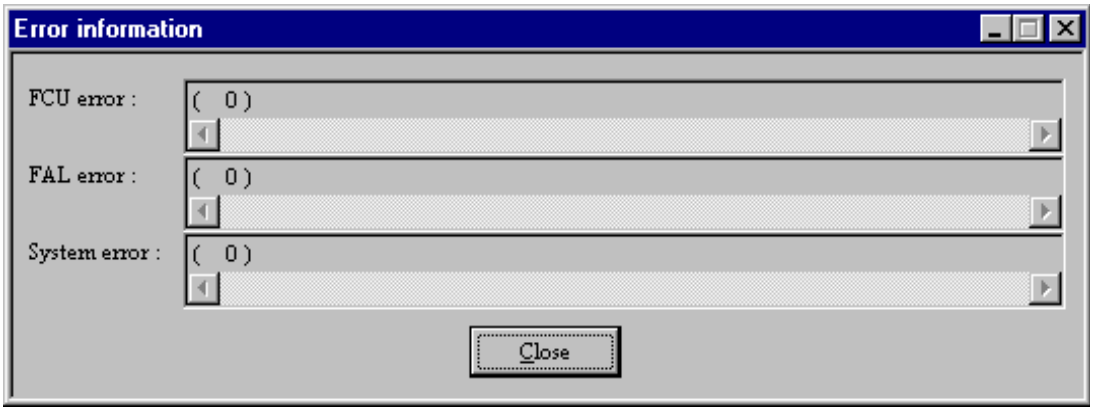


Figure 5.12 Error Information Panel

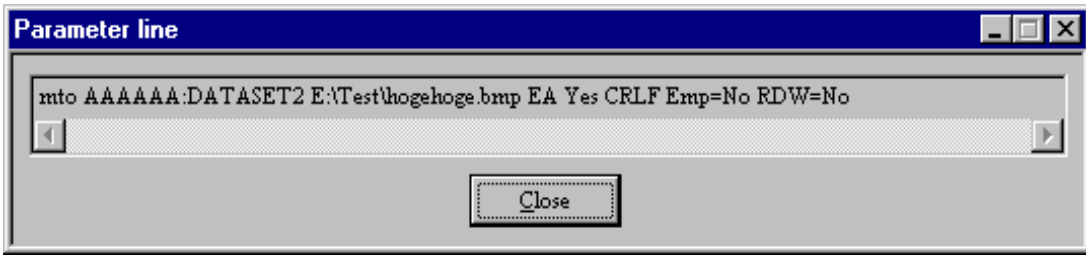


Figure 5.13 Parameter Line Panel

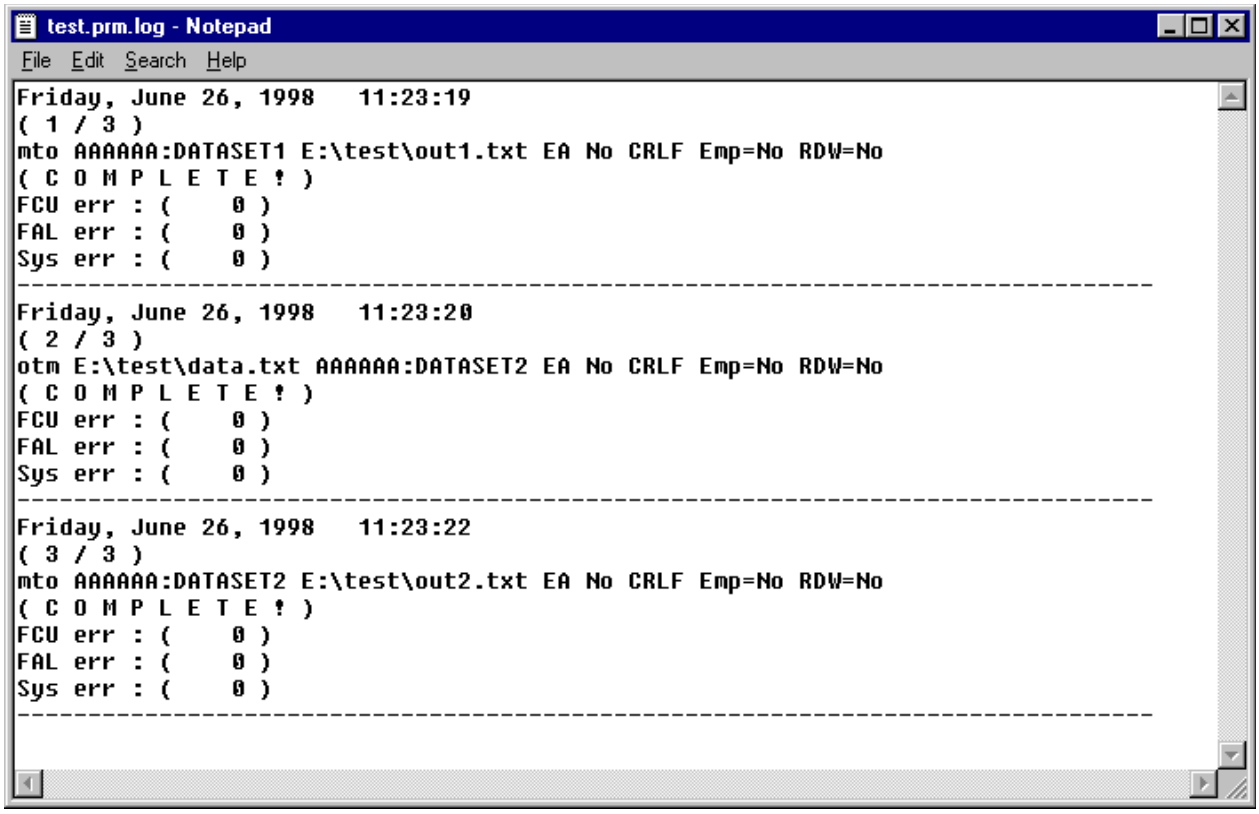


Figure 5.14 FCU Log File Display in Notepad

5.2.3 Creating an FCU Parameter Definition File Using FCU for Windows NT

To create an FCU parameter definition file using the FCU GUI for Windows NT:

1. Start the FCU GUI for Windows NT (see section 5.2.1). If you start FCU from the DOS prompt, enter **fcu** (do not specify the **-nc** or **param** option).
2. When the FCU main panel opens (see section 5.2.2), the title bar should display **Untitled** to indicate that a new parameter definition file is open. If a file name is displayed instead of **Untitled**, use the **File-New** command ( ) to open a new parameter definition file.
3. If you plan to perform HMDE operations while you are creating the FCU parameter definition file, open the Volume information panel (select **View-Volume information…**), and make sure that the desired HMDE volume(s) is/are available. If not, FCU will not be able to perform HMDE operations, but you can still create a new parameter definition file.
4. Enter the desired FCU initiation parameters for the first HMDE operation.
   - Use the buttons to select the transfer direction (**M to O** or **O to M**).
   - Enter the source and target datasets/files in the **Mainframe File** field (VSN:dataset), and/or **Open-system file** field (/directory/…/filename).
   - Open the Option panel (refer to Figure 5.11) using the **View-Option…** menu command ( ), and then select the desired FCU options (code conversion, padding, delimiters, etc.). Do not select **Continuous execution** or **Clear log file** when creating a new FCU parameter definition file. Close the Option panel when you are done.
5. When the FCU initiation parameters are correct, select the **Parameter-Save-Insert** command ( ) to save the current parameter set in the new FCU parameter definition file. The status bar now displays 1/1 to indicate that line one of one is now being displayed. If the HMDE volume is available, you can perform the operation now by clicking **Execute**. If the **Execute** button is not enabled, the parameter set has not been saved in the file.
6. Repeat steps (4) and (5) to add each new FCU initiation parameter set to the new FCU parameter definition file. If desired, you can use the **Parameter-Wipe** menu command ( ) to clear the screen before you enter the next set of parameters, or you can leave the parameters and make changes where needed to specify the next new line in the file. Make sure to keep HMDE operations which use OPEN-x FMT volumes in a separate FCU parameter definition file from operations which use -A, -B, and -C volumes.
7. If you need to insert a new line between two existing lines, go to the line above/before the line to be inserted using the **Parameter-Load** commands ( ), change the parameters as needed, and then use the **Parameter-Save-Insert** command ( ) to insert the new line. The new line is inserted below/after the current line number.
8. If you need to modify an existing line, go to the line to be modified using the **Parameter-Load** commands, change the parameters as needed, and then use the **Parameter-Save-Replace** command ( ) to modify the line as specified.
9. When you want to save your new FCU parameter definition file, select the **File-Save** menu command ( ). The file extension must be **.prm**.

5.2.4 Performing HMDE Operations Using the FCU GUI for Windows NT

To perform HMDE file transfer operations using the FCU GUI for Windows NT:

1. If you will be performing HMDEmto operations:
   a) Make sure that the source datasets are located on the desired HMDE volume(s). If you will not be using an existing FCU parameter definition file, write down the VSN:dataset of the source dataset and the complete path and file name of the target file for each HMDEmto operation.
   b) Verify that the HMDEmto target files do not already exist (or can be overwritten).
   c) Vary the HMDEmto volume(s) and channel path(s) offline from the S/390 host.
2. If you will be performing HMDEotm operations:
   a) Make sure that the source files are located on the desired HMDE volume(s). If you will not be using an existing FCU parameter definition file, write down the complete path and file name of the source file and the VSN:dataset of the target dataset for each HMDEotm operation.
   b) Create and allocate the target datasets. This ensures that the target dataset is registered in the VTOC. Make sure to allocate enough space and to use the appropriate record format and record length for the data to be transferred.
   c) Vary the HMDEotm volume(s) and channel path(s) offline from the S/390 host.
3. If you will be performing HMDEoto operations:
   a) If you will not be using an existing FCU parameter def. file, write down the complete path and file name of the source and target files for each HMDEotm/mto operation.
   b) Use the ALC utility to allocate the intermediate datasets on the HMDEoto volume(s) (see section 4.3). Make sure to allocate enough space and to use the appropriate record format and record length for the data to be transferred.
   c) Verify that the HMDEoto target files do not already exist (or can be overwritten).
4. Make sure that the desired HMDE volume definition file (HMDEoto only, or HMDEmto and HMDEotm) is available for use by FCU (**datasetmount.dat** in current directory).
5. Start FCU (see section 5.2.1). If you want to specify any FCU options, start FCU from the DOS prompt. ***Note:*** If you specify the **-nc** option, FCU performs all specified operations continuously, then self-terminates and displays any error information at the DOS prompt.
6. When the FCU main panel opens, select the **View-Volume information…** command () to open the **Volume information** panel, and verify that the desired HMDE volume(s) is/are available. If not, edit the HMDE volume definition file as needed (see section 4.1).
7. Make sure that the desired FCU parameter definition file is open (displayed in title bar). If not, open the desired FCU parameter definition file using the **File-Open** command (). If you want to create a new file using the FCU GUI, see section 5.2.3 for instructions.
8. The FCU main panel displays the first/next parameter set in the specified FCU parameter definition file. If you want to perform this HMDE operation, click **Execute**. If not:
   a) You can load the next parameter set using the **Parameter-Load-Next** command ().
   b) You can delete the current parameter set from the FCU parameter definition file using the **Parameter-Delete** command (). The next parameter set loads automatically.

c) You can modify the current parameter set as follows: change the FCU parameters as needed, and then use the **Parameter-Save-Replace** command () to replace the previously loaded parameter set with the new parameter set.

d) You can add a new parameter set as follows: change the FCU parameters as needed, and use the **Parameter-Save-Insert** command () to insert the new parameter set below the current parameter set.

e) If the **Continuous operation** option is selected, you can open the Error information panel before starting the operations to monitor the FCU processing. Move the Error information panel so that it does not overlap the FCU main panel.

9. When the desired HMDE operation is displayed, click **Execute** to start the operation. (If the **Execute** button is not enabled, you have not saved the current parameter set.) If the **Continuous operation** option was selected, FCU will process all operations from the current line to the end of the file and then self-terminate. The error information for these operations is placed in the most recent **.log** file(s) in the current directory (e.g., **mto.log**).

   WARNING: In some early versions of FCU for Windows NT, FCU may overwrite existing Windows NT target files without requesting confirmation.

10. If you started an HMDEmto operation and the target file already exists, FCU requests overwrite confirmation. Click **OK** to overwrite the target file, or click **Cancel** to cancel the operation.

11. When FCU starts the operation, the Execute panel opens and displays the progress of the operation. To cancel the operation in progress, select **Cancel**.

    Note: The Execute dialog panel will not appear when the mainframe OS is VSE.

12. When the operation is complete, the Execute panel displays the result (see Figure 5.15 through Figure 5.17). If an error occurred, the Error information panel opens automatically to display the error (refer to Figure 5.12). See Appendix C for further information on errors.

13. FCU does not load the next operation automatically. To perform another HMDE operation, select the desired **Parameter-Load** command, and repeat steps (8) through (12). To exit FCU, select the **File-Exit** command.

Note: After an HMDEotm file transfer from Windows NT, there will be a delay before you can access the HMDE volume. The length of delay varies according to individual server performance.

Note: After you expand open volumes (LUSE), you will need to reboot Windows NT.

Note: Do not use the open system host to access an HMDE volume. Use only FAL to access HMDE volumes. This applies to PC server platforms (e.g., NT) and UNIX-based systems.

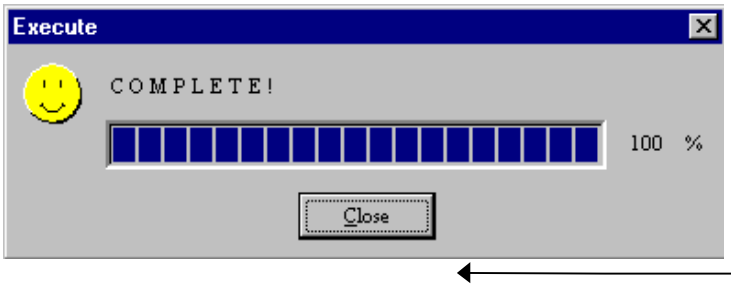


Figure 5.15 Execute Panel Showing Normal End

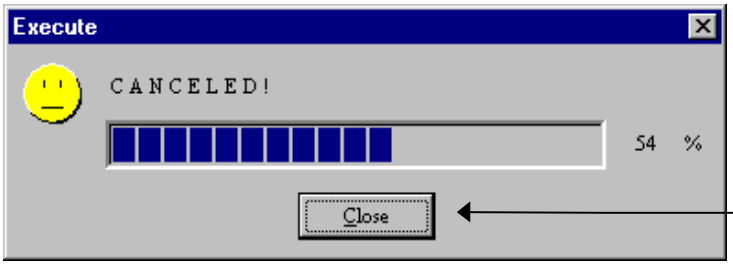


Figure 5.16 Execute Panel Showing Canceled Operation

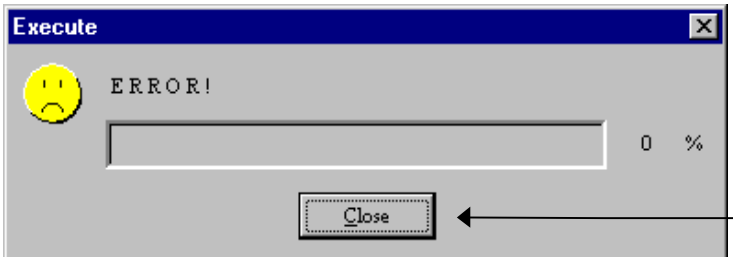


Figure 5.17 Execute Panel Showing Error End

Chapter 6 Using the File Access Library (FAL)

The FAL component of HMDE consists of the object module file **fal.o** (**fal.obj** for Windows NT) and the header file **dataset.h**. The FAL provides several important C functions (Visual C++ for Windows NT) which enable user applications on the open-system hosts to access S/390 data on the 9900 HMDE volumes.

6.1 FAL Requirements

The FAL functions have the same dataset requirements as FCU (e.g., SAM, standard R0 track format) (see section 4.2). The FAL also has the following additional requirements:

- The FAL functions support only standard MVS VTOC. The FAL functions cannot access MVS datasets managed by an index VTOC and cannot access VSE datasets when called from user applications.
- The FAL functions are not "thread-safe." The FAL functions may not operate properly when used by multiple threads within a single process.
- The FAL functions cannot be used by a "signal handler." If this accidentally happens and memory space is occupied, use **kill** to cancel the processes, and use **ipcrm** to delete the shared memory areas that have KEY=0 (refer to OS manuals). Do not issue the following signals to an FAL/FCU process (UNIX only):
  SIGUSR1, SIGUSR2, SIGILL, SIGTRAP, SIGIOT, SIGABRT, SIGEMT, SIGFPE, SIGKILL, SIGBUS, SIGSEGV, SIGSYS, SIGALRM, SIGPOLL, SIGIO, SIGSTOP, SIGTSTP, SIGCONT, SIGTTIN, SIGTTOU, SIGVTALRM, SIGPROF, SIGXCPU, SIGXFSZ, SIGWAITING, SIGLWP, SIGFREEZE, SIGTHAW, SIGCANCEL
- The following terminology is reserved for the FAL functions and cannot be used in function names, variable names, or constant symbols in the user application:
  - Words that begin with **dataset** or **fast_**
  - **GetVolSers**

6.2 FAL Functions

The FAL includes the following C functions (Visual C++ for Windows NT):

- Opening a dataset: **datasetOpen** (see section 6.2.1)
- Reading one record from a dataset: **datasetGet** (see section 6.2.2)
- Writing one record to a dataset: **datasetPut** (see section 6.2.3)
- Closing a dataset: **datasetClose** (see section 6.2.4)
- Acquiring error information: **datasetGetLastError** (see section 6.2.5)
- Acquiring dataset attribute information (see section 6.2.6):
  datasetGetFileInformation **datasetFindNextFile**
  datasetFindFirstFile **datasetFindClose**
- Converting dataset attribute information (see section 6.2.7):
  datasetGetDsorgString **datasetGetRecfmString**

6.2.1 Opening a Dataset

datasetHandle = datasetOpen (pathname, mode)

The **datasetOpen** function opens the dataset specified by **pathname** for the type of access specified by **mode**. Table 6.1 shows the **datasetOpen** arguments and return values.

Table 6.1 DatasetOpen Function

Item	Value	Type	Description
Argument	pathname	char *	VSN:Dataset name VSN = 6-character volser. Volume must be listed in HMDE volume definition file. Delimiter = : (colon, no spaces allowed) Dataset name: 44 characters max, no spaces allowed.
	mode	char *	r = open dataset for read access w = open dataset for write access
Return value	datasetHandle	DATASET_HANDLE	Handle
	-1		Error end

When the **datasetOpen** function terminates successfully, it returns a handle which identifies the dataset opened. The **datasetHandle** information is used as the argument of subsequent functions such as **datasetGet**, **datasetPut**, or **datasetClose**. Do not change the **datasetHandle** value returned by this function. If the **datasetOpen** function terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5).

The **datasetOpen** function has the following restrictions:

- Only one dataset at a time can be open within one process. When multiple datasets need to be opened, the open dataset must be closed before another dataset can be opened. This restriction does not apply to open-system files.
- A dataset which is being accessed by the **datasetFindFirstFile** or **datasetFindNextFile** function cannot be opened. The **datasetFindClose** function must be executed before the dataset can be opened. This restriction does not apply to open-system files.

6.2.2 Reading Data

reclen = datasetGet (datasetHandle, buf, buflen)

The **datasetGet** function reads one record from the specified dataset (**datasetHandle**) and puts the record into a buffer (**buf**) of length **buflen**. The **datasetGet** function extracts only the data entity from each record and does not transfer the BL and RL bytes for variable-length records to the buffer. Table 6.2 shows the **datasetGet** arguments and return values.

Table 6.2 DatasetGet Function

Item	Value	Type	Description
Argument	datasetHandle	DATASET_HANDLE	The datasetHandle value returned by the **datasetOpen** function.
	buf	char *	Buffer area for storing the read data.
	buflen	long	Size of the buffer area. If the record is larger than **buflen** or equal to zero, **datasetGet** returns an error and does not transfer any data to the **buf**.
Return value	reclen	long	Data entity size transferred to the buffer
	-1		Error end

Figure 6.1 shows the format requirements for variable-length records accessed by the **datasetGet** function. Each variable-length block must start with the two-byte BL field, and each variable-length record must start with the two-byte RL field. The **datasetGet** function automatically extracts the data entities without the BL and RL fields.

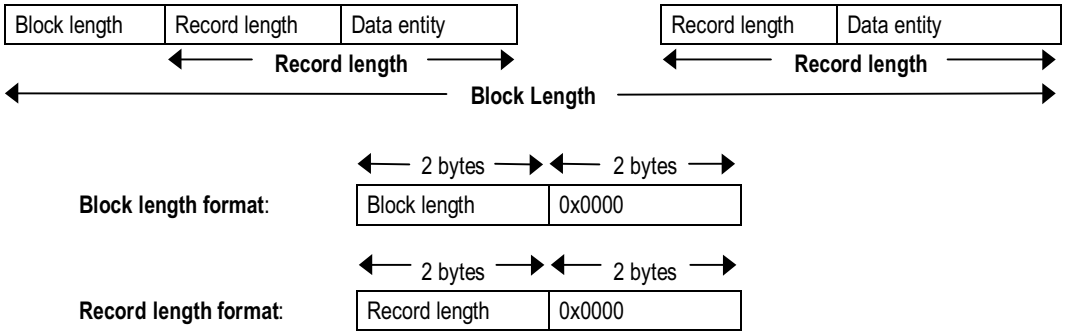


Figure 6.1 Format Requirements for Reading Variable-Length Records

When the **datasetGet** function terminates successfully, it returns the length of the data entity read from the dataset. If the **datasetGet** function detects the end of dataset (EOF) or terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section6.2.5). For example, when the **datasetGet** function detects EOF, **datasetGetLastError** will return **DATASET_ERROR_END_OF_FILE**.

6.2.3 Writing Data

reclen = datasetPut (datasetHandle, buf, buflen)

The **datasetPut** function writes one record from the **buf** into the dataset specified by **datasetHandle**. Table 6.3 shows the **datasetPut** arguments and return values.

Table 6.3 DatasetPut Function

Item	Value	Type	Description
Argument	datasetHandle	DATASET_HANDLE	The datasetHandle value returned by the **datasetOpen** function.
	buf	char *	Buffer area for storing the write data.
	buflen	long	Size of the buffer area. If any of the following conditions is detected, **datasetPut** returns an error and does not transfer any data to the dataset: For fixed-length record: **buflen** ≠ RL of the dataset For variable-length record: (**buflen** + 4) > RL of dataset For variable-length record: **buflen** = 0 (no data entity)
Return value	reclen	long	Data entity size written into the dataset.
	-1		Error end

Figure 6.2 shows the format requirements for variable-length records accessed by the **datasetPut** function. When the target dataset is variable-length, the **datasetPut** function takes the data entity from the **buf**, automatically adds the two-byte RL field, and writes the record into the dataset. When the data is written into the dataset, multiple records are blocked within the extent defined by the VTOC of the dataset.

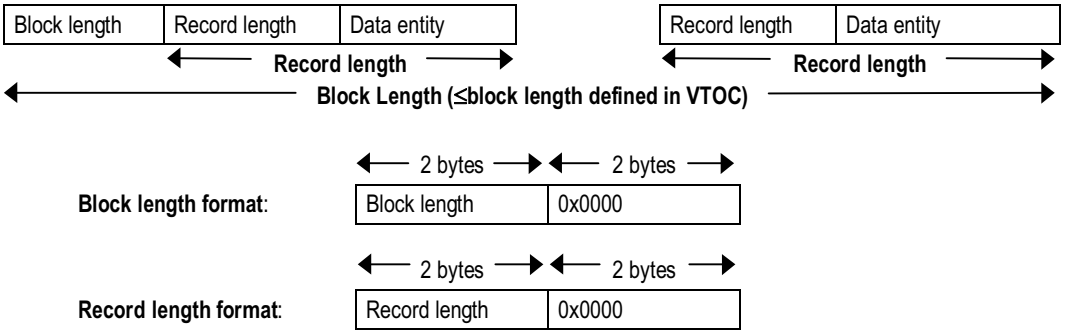


Figure 6.2 Format Requirements for Writing Variable-Length Records

When the **datasetPut** function terminates successfully, it returns the length of the data entity written to the dataset. If the **datasetPut** function terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5).

6.2.4 Closing a Dataset

datasetError = datasetClose (datasetHandle)

The **datasetClose** function closes the dataset specified by **datasetHandle**, which is returned by the **datasetOpen** function. Table 6.4 shows the **datasetClose** arguments and return values.

Table 6.4 DatasetClose Function

Item	Value	Type	Description
Argument	datasetHandle	DATASET_HANDLE	The datasetHandle value returned by the **datasetOpen** function.
Return value	0 -1		Normal end Error end

When the **datasetClose** function terminates successfully, it returns a value of 0. If it terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5).

6.2.5 Acquiring Error Information

datasetLastError = datasetGetLastError()

The **datasetGetLastError** function acquires the error code information for the most recent error. Errors in FAL functions are defined in **dataset.h** (see Appendix C). Errors in UNIX are defined by a standard error file (**errno.h**). Errors in Windows NT are defined by **errno.h** attached with Microsoft Visual C++. Table 6.5 shows the **datasetClose** arguments and return values.

Table 6.5 DatasetGetLastError Function

Item	Value	Type	Description
Argument	none	--	--
Return value	datasetLastError	Long	Error code

6.2.6 Acquiring Dataset Attributes

FAL provides several functions for acquiring the complete dataset attribute information for one or more datasets: **datasetGetFileInformation**, **datasetFindFirstFile**, **datasetFindNextFile**, and **datasetFindClose**. The dataset attribute information returned by these functions contains:

```
typedef struct DATASET_FIND_DATA {
   unsigned short blockSize;                /* Block length    */
   unsigned short recordSize;               /* Record length   */
   unsigned char dsorg[2];                  /* dataset type    */
   unsigned char recfm;                     /* record format   */
   char name[44];                           /* dataset name    */
   unsigned short lastBlockTt;              /* last block address (relative track number) */
   unsigned char lastBlockR;                /* last block address (relative record number)*/
}  DATASET_FIND_DATA;
```

6.2.6.1 Acquiring attribute information for a specific dataset

datasetError = datasetGetFileInformation (pathname, &ffd)

The **datasetGetFileInformation** function acquires the attribute information for the dataset specified by **pathname** and returns the data into **ffd**. Table 6.6 shows the **datasetGetFileInformation** arguments and return values.

Table 6.6 DatasetGetFileInformation Function

Item	Value	Type	Description
Argument	pathname	char *	VSN:Dataset name VSN = 6-character volser. Volume must be listed in volume definition file. Delimiter = : (colon, no spaces) Dataset name: 44 characters max, no spaces.
	ffd	DATASET_FIND_DATA	Area where the dataset attribute information is stored.
Return value	0		Normal end
	-1		Error end

When the **DatasetGetFileInformation** function terminates successfully, it returns a value of 0. If it terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5).

The **DatasetGetFileInformation** function has the following restriction:

- The **DatasetGetFileInformation** function cannot be used on an open dataset. Use this function before opening or after closing the dataset.

6.2.6.2 Acquiring attribute information for multiple datasets

A combination of the **datasetFindFirstFile**, **datasetFindNextFile**, and **datasetFindClose** functions is used to acquire attribute information for more than one dataset in the specified S/390 volume.

1. datasetHandle = datasetFindFirstFile (pathname, &ffd)

The **datasetFindFirstFile** function acquires the attribute information for the first dataset in the volume specified by **pathname** and returns the data into **ffd**. Table 6.7 shows the **datasetFindFirstFile** arguments and return values.

Table 6.7 DatasetFindFirstFile Function

Item	Value	Type	Description
Argument	pathname	char *	VSN VSN = 6-character volser. Volume must be listed in volume definition file.
	ffd	DATASET_FIND_DATA	Area where the dataset attribute information is stored.
Return value	datasetHandle	DATASET_HANDLE	Normal end
	-1		Error end

When the **datasetFindFirstFile** function terminates successfully, it returns a handle which identifies the dataset for which the attribute information was acquired. The **datasetHandle** information is used as the argument of the subsequent functions **datasetFindNextFile** and **datasetFindClose**. Do not change the **datasetHandle** value returned by this function. If the **datasetFindFirstFile** function terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5). For example, when the **datasetFindFirstFile** function does not find any datasets in the VTOC, the **datasetGetLastError** function will return **DATASET_ERROR_END_NO_DATASET**.

The **datasetFindFirstFile** function has the following restrictions:

- The **datasetFindFirstFile** function cannot be used on an open dataset. Use this function before opening or after closing the dataset.
- After a dataset has been accessed by the **datasetFindFirstFile** function, the dataset cannot be opened until after the **datasetFindClose** function is called.

2. datasetError = datasetFindNextFile (datasetHandle, &ffd)

The **datasetFindNextFile** function acquires the attribute information for the next dataset in the volume specified by **datasetHandle** and returns the data into **ffd**. This function can be repeated until "no dataset found" is returned, or until the user application determines that no more information is needed. Table 6.8 shows the **datasetFindNextFile** arguments and return values.

Table 6.8 DatasetFindNextFile Function

Item	Value	Type	Description
Argument	datasetHandle	DATASET_HANDLE	The datasetHandle value returned by the preceding **datasetFindFirstFile** function.
	ffd	DATASET_FIND_DATA	Area where the dataset attribute information is stored.
Return value	0		Normal end
	-1		Error end, or no dataset found

When the **datasetFindNextFile** function terminates successfully, it returns a value of 0. If this function terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5). For example, when the **datasetFindNextFile** function cannot find the next dataset in the VTOC, the **datasetGetLastError** function will return **DATASET_ERROR_END_OF_VTOC**.

The **datasetFindNextFile** function has the following restrictions:

- The **datasetFindNextFile** function cannot be used on an open dataset. Use this function before opening or after closing the dataset.
- After a dataset has been accessed by the **datasetFindNextFile** function, the dataset cannot be opened until after the **datasetFindClose** function is called.
- The **datasetFindFirstFile** function must be called prior to **datasetFindNextFile**.

3. datasetError = datasetFindClose (datasetHandle)

The **datasetFindClose** function terminates the acquisition of attribute information by the **datasetFindFirstFile** and **datasetFindNextFile** functions and closes the dataset. The **datasetFindFirstFile** function must be called prior to **datasetFindClose**. Table 6.9 shows the **datasetFindClose** arguments and return values.

Table 6.9 DatasetFindClose Function

Item	Value	Type	Description
Argument	datasetHandle	DATASET_HANDLE	The datasetHandle value returned by the preceding **datasetFindFirstFile** function.
Return value	0		Normal end
	-1		Error end

When the **datasetFindClose** function terminates successfully, it returns a value of 0. If this function terminates unsuccessfully, it returns a value of -1. To get the error code information, execute the **datasetGetLastError** function (see section 6.2.5).

6.2.7 Converting DO and RF Information

The FAL provides two functions for converting specific attribute information from a dataset into character strings: **datasetGetDsorgString**, and **datasetGetRecfmString**.

6.2.7.1 Converting the dataset organization (DO) type value

datasetError = datasetGetDsorgString (dsorg, text)

The **datasetGetDsorgString** function converts the dataset organization (DO) type to a three-byte character string. The DO type is specified by **dsorg[2]** in **DATASET_FIND_DATA**. Table 6.10 lists the **datasetGetDsorgString** arguments and return values.

Table 6.10 DatasetGetDsorgString Function

Item	Value	Type	Description
Argument	dsorg	u_char[ ]	Value of dsorg[2] (two bytes) obtained by the attribute acquisition function **datasetGetFileInformation**, **datasetFindFirstFile**, or **datasetFindNextFile**.
	text	char [3]	Character string indicating the dataset organization (DO) type (3 bytes): PS physical sequential organization VS VSAM organization DA direct access organization PO Partial organization ** Other than above types
Return value	0		Normal end
	-1		Error end

6.2.7.2 Converting the Record Format (RF) Type Value

datasetError = datasetGetRecfmString (recfm, text)

The **datasetGetRecfmString** function converts the record format (RF) type to a five-byte character string. The RF type is specified by **recfm** in **DATASET_FIND_DATA**. Table 6.11 lists the **datasetGetRecfmString** arguments and return values.

Table 6.11 DatasetGetRecfmString Function

Item	Value	Type	Description
Argument	recfm	u_char	Value of recfm (one byte) obtained by attribute acquisition function **datasetGetFileInformation**, **datasetFindFirstFile**, or **datasetFindNextFile**.
	text	char [5]	Character string (5 bytes) indicating the (RF) type: text[0] F fixed-length record V variable-length record U unknown-length record text[1] B blocking record sp spanned record st standard format record
Return value	0		Normal end
	-1		Error end

6.3 Using the FAL Functions

The FAL functions can be executed by any C program on the UNIX host. The FAL does not support C++. The S/390 datasets accessed by the FAL functions must be located on 9900 HMDE volumes. The HMDE volumes must be installed and configured correctly (see section 2.2), the FAL/FCU software must be installed properly (see section 2.3), and the HMDE volume definition file must be available and configured correctly (see section 4.1). Since FAL operations do not involve GUI windows, the X windows environment and FcuMf resource file are not required.

Figure 6.3 shows an example of reading data using the FAL functions. Figure 6.4 shows an example of acquiring attribute information using the FAL functions. To use the FAL functions in a C program (Visual C++ for Windows NT):

1. Copy the HMDE volume definition file (**datasetmount.dat**) to the directory containing the C program that will call the FAL C function(s).
2. Include the FAL header file (**dataset.h**) within the C program that will call the FAL function(s) (e.g., copy **dataset.h** to **/usr/include**).
3. Compile the C program as follows:

   IBM AIX
   # cc -qlanglvl=ansi -o Output file name Source file name /usr/lib/libfal.a
   libfal.a = file name of FAL object module

   HP-UX
   # cc -Ae +DAportable -o Output file name Source file name /usr/lib/libfal.sl
   libfal.sl = file name of FAL object module

   Sun Solaris
   # cc -o Output file name Source file name /usr/lib/libfal.so.1
   libfal.so.1 = file name of FAL object module

   DIGITAL UNIX
   # c89 -o Output file name Source file name /usr/lib/libfal.so
   libfal.so = file name of FAL object module

   NCR SVR4 UNIX
   # cc -o Output file name Source file name /usr/lib/libfal.so
   libfal.so = file name of FAL object module

   Sequent DYNIX/ptx
   # cc -lseq -o Output file name Source file name /usr/lib/libfal.so
   libfal.so = file name of FAL object module

   Windows NT (Visual C++)
   a) Start **Developer Studio**, and create a new project.
   b) Copy the following three FAL files into the project folder: **dataset.h**, **fal.dll**, **fal.lib**
   c) Select **Settings** in the **Projects** menu of **Developer Studio**.
   d) On the Project Settings panel select the **Link** tab.
   e) Enter **fal.lib** in the **object/library module** field.
   f) Build and execute.

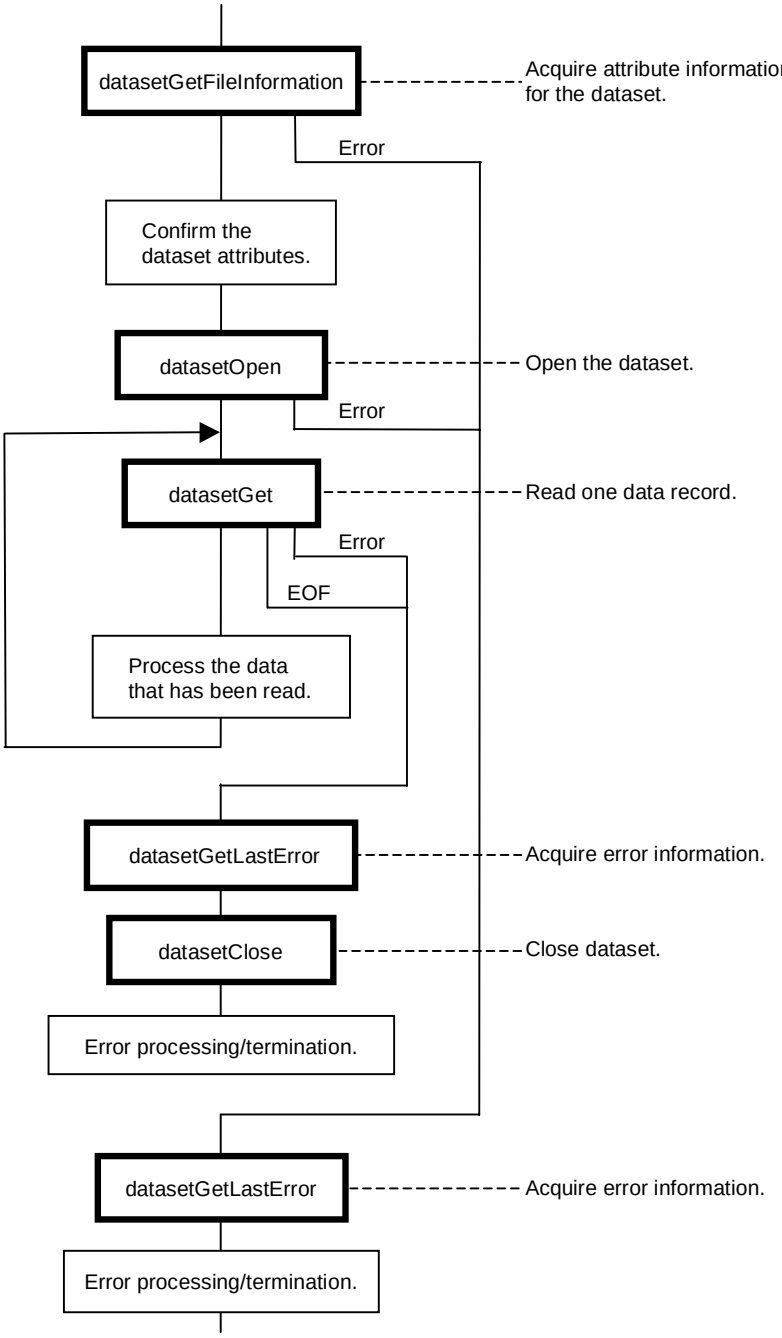


Figure 6.3 Example of Reading Data from an S/390 Dataset Using FAL

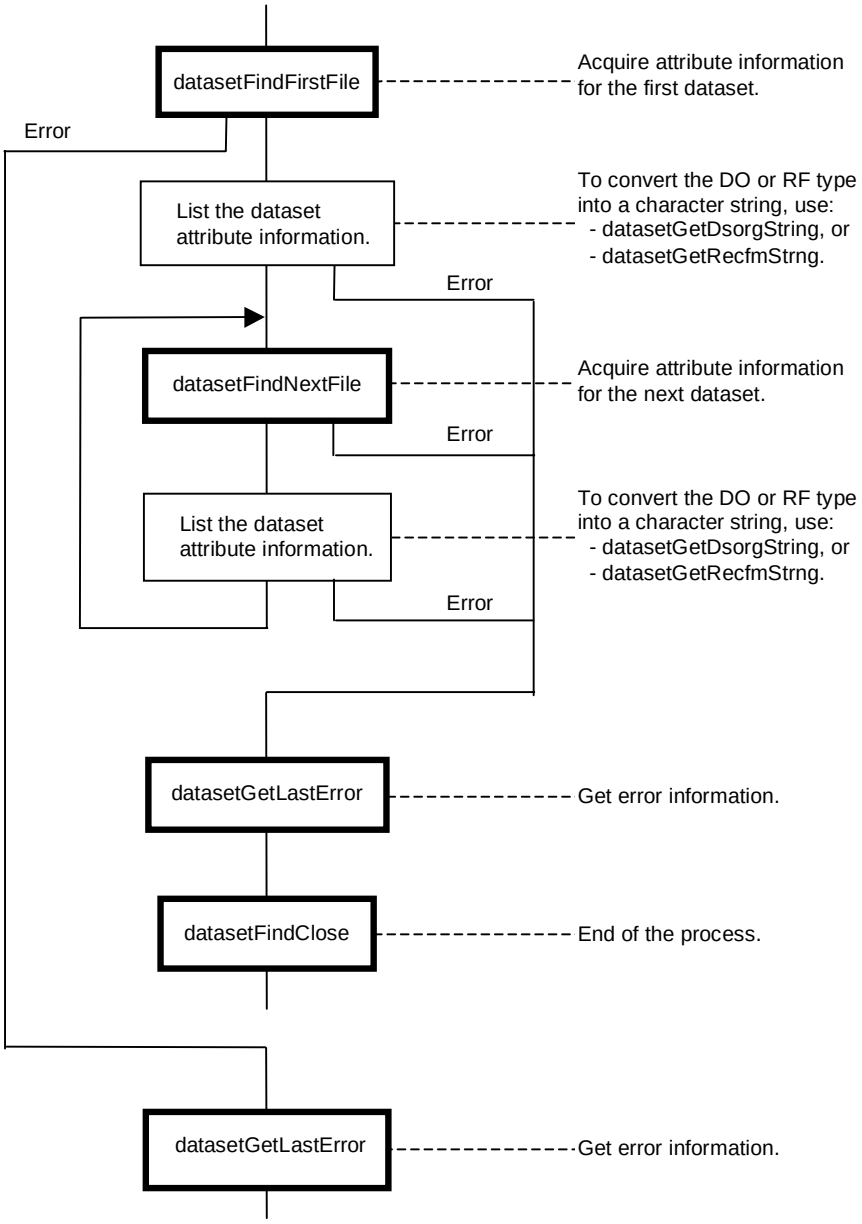


Figure 6.4 Example of Acquiring S/390 Dataset Attributes Using FAL

Chapter 7 Troubleshooting

7.1 Troubleshooting

For general troubleshooting information on the 9900 subsystem, please refer to the *Hitachi Lightning 9900™ User and Reference Guide* (MK-90RD008).

The FAL/FCU software is not expected to fail in any way. When errors are detected, error codes and messages are displayed and/or logged. Appendix C lists the FAL and FCU error codes and provides instructions for resolving the error conditions.

If you have a problem with the FAL/FCU software, first make sure that the problem is not being caused by other open-system software or hardware, and try rebooting the open-system server. For FCU operations, make sure that the HMDE volume definition file and FCU initiation parameters are correct. Table 7.1 lists potential error conditions in FAL/FCU and provides instructions for resolving each condition. If you are still unable to resolve an error condition, please ask your Hitachi Data Systems representative for help, or call the Hitachi Data Systems Support Center for assistance (see section 7.2).

Table 7.1 Troubleshooting

Error Condition	Recommended Action
UNIX files in non-9900 subsystem could not be accessed.	Make sure that the devices have been mounted. If mounting is done during an FCU operation, the results cannot be guaranteed because error information may not be reported to FCU.
Sun Solaris system reports an error indicating **libXm.so.xx is not found**.	Define a path to the Xmlibrary as follows: 1. For C shell, add the following line to the **.cshrc** file in the home directory: **setenv LD_LIBRARY_PATH /usr/dt/lib:$LD_LIBRARY_PATH** 2. For non-C shell, add the following two lines to the **.dtprofile** file in the home directory: **LD_LIBRARY_PATH=/usr/dt/lib:$LD_LIBRARY_PATH** **export LD_LIBRARY_PATH**
Windows NT only: FCU reports errors when accessing an FCU parameter definition file.	Remove all space lines from the FCU parameter definition files. FCU versions 01-01-24 and earlier supported space lines when run without the GUI. FCU versions 01-01-36 and later do not support space lines.
FCU reports code conversion table errors.	If you specified your own code conversion table, make sure that the file name and path are correct. FCU may also report code conversion table errors when the HMDE volume definition file contains both mainframe and OPEN-xHMDE volumes. Keep the HMDEoto volume definition file separate from the HMDEmto/otm volume definition file.

7.2 Calling the Hitachi Data Systems Support Center

If you need to call the Hitachi Data Systems Support Center, make sure to provide as much information about the problem as possible, including the circumstances surrounding the error or failure and the exact content of any messages displayed on the S/390 and/or open-system hosts. Please collect the following information before you call for technical support:

- Error codes: FCU error code, FAL error code, SYS error code. Use the FCU GUI to check recent error information (**Help-Error** command for UNIX, **View-Error information…** command for NT).
- FCU parameters: direction (mto or otm), input and output files, and FCU options (code conversion, padding, delimiter, empty file, RDW, VSE record).
- HMDE volume definition file: contents
- FCU parameter definition file (if used): contents
- Command line log (if possible).
- FAL error logs. The FAL logs for UNIX are **/tmp/fal_error** and **/tmp/fal_error.bak**, and **/tmp/fal_dump** and **/tmp/fal_dump.bak**. The FAL logs for Windows NT are **c:\fal_error** and **c:\fal_error.bak**, and **c:\fal_dump** and **c:\fal_dump.bak**.
- Windows NT only: FCU log file (e.g., **fcudata.prm.log**), and Dr. Watson's log file (e.g., **c:\WINNT\DRWTSN32.LOG**).
- **Syslog**: error information and other applicable contents

The worldwide Hitachi Data Systems Support Centers are:

- Hitachi Data Systems North America/Latin America
  San Diego, California, USA
  1-800-348-4357
- Hitachi Data Systems Asia Pacific
  North Ryde, Australia
  011-61-2-9325-3300
- Hitachi Data Systems Europe
  Contact Hitachi Data Systems Local Support

Appendix A Acronyms and Abbreviations

ASCII	American National Standard Code for Information Interchange
BL	block length
CKD	count key data
CR	carriage return
D	delimiter
DAM	direct-access method (not supported by HMDE)
DAT	digital audio tape
DO	dataset organization
DS	dataset size
EA	EBCDIC/ASCII (includes EBCDIC-to-ASCII and ASCII-to-EBCDIC)
EBCDIC	extended binary-coded decimal interchange code
Emp	empty file option
EOF	end of file
ESCON	Enterprise System Connection (IBM trademark for optical channels)
F	fixed-length and de-blocking (S/390 record format)
FAL	File Access Library
FB	fixed-length and blocking (S/390 record format)
FC	fibre channel
FCU	File Conversion Utility
FD	floppy disk
FWD	fast-wide differential
GUI	graphical user interface
HMBR	Hitachi Multiplatform Backup/Restore
HMDE	Hitachi Multiplatform Data Exchange
ICKDSF	A Device Support Facilities (DSF) command used to perform media maintenance
kB	kilobytes
LDEV	logical device
LF	line feed
LSM	Logical Storage Manager
LU	logical unit
LUN	logical unit number
MF	mainframe
mto	mainframe-to-open
MVS	Multiple Virtual Storage
OPEN-x	standard LU type, e.g., OPEN-3, OPEN-9
OS	operating system

otm	open-to-mainframe
oto	open-to-open
PAM	partitioned access method (not supported by HMDE)
R0	record 0
RDW	record description word
RF	record format
RL	record length
SAM	sequential-access method, System Administration Manager (HP-UX)
SCSI	small computer system interface
SMIT	System Management Information Tool (IBM AIX)
V	variable-length and de-blocking (S/390 record format)
VB	variable-length and blocking (S/390 record format)
volser	volume serial number
VOS3	Virtual-Storage Operating System 3 (a Hitachi S/390 OS)
VSAM	Virtual Storage Access Method (not supported by HMDE)
VSE	Virtual Storage Extended
VSN	volume serial number
VTOC	volume table of contents
WinNT	Windows NT

Appendix B Using FCU Without the GUI

B.1 Using FCU Without the GUI

FCU can be used without the GUI to perform HMDE operations. To use FCU without the GUI, you must start FCU using the **-nw** option. The FCU options are:

- The **-nw** option (**nw** = no window) tells FCU to execute the specified HMDE operation(s) without displaying the GUI. When you use this option (entered as **fcunw** or **fcu -nw**), FCU uses the FCU initiation parameters specified by the **param** option to perform HMDE operations. To use FCU without the GUI, the **-nw** option is required.

  Note: FCU versions 01-01-36 and later for Windows NT do not support the **-nw** option.

- The **-nc** option (**nc** = no checking) tells FCU to execute all specified HMDE operations without requesting confirmation for FCU parameters or existing HMDEmto target files. If you want to bypass these confirmations, enter **-nc**. FCU will perform all specified operations and overwrite existing HMDEmto target files. If you want to check the FCU parameters and the HMDEmto target file before starting each operation, do not enter **-nc**.

- The **param** option (param = FCU parameter definition file) tells FCU whether to use an FCU parameter definition file or a specific FCU initiation parameter set to perform HMDE operations. The **param** option must have one of the following three values:

  – [blank]. If you want to use the default FCU parameter definition file (**fcudata.param** located in the current directory) to perform HMDE operations, leave the **param** option blank (do not enter anything).

  – **file_name**. If you want to use a different FCU parameter definition file to perform HMDE operations, enter the filename of the file. Make sure to enter the complete path (absolute or relative path) if the file is not in the current directory.

  – **-P + parameters**. If you want to perform one specific HMDE operation, enter **-P** followed by the FCU initiation parameter set (e.g., **mto VSN:dataset targetfile No No No**) for the desired HMDE operation.

Note: FCU for UNIX cannot be used by a "signal handler." If this accidentally happens and memory space is occupied, use **kill** to cancel the processes, and use **ipcrm** to delete the shared memory areas that have KEY=0 (refer to OS manuals). Do not issue the following signals to an FCU process (UNIX only):

SIGUSR1, SIGUSR2, SIGILL, SIGTRAP, SIGIOT, SIGABRT, SIGEMT, SIGFPE, SIGKILL, SIGBUS, SIGSEGV, SIGSYS, SIGALRM, SIGPOLL, SIGIO, SIGSTOP, SIGTSTP, SIGCONT, SIGTTIN, SIGTTOU, SIGVTALRM, SIGPROF, SIGXCPU, SIGXFSZ, SIGWAITING, SIGLWP, SIGFREEZE, SIGTHAW, SIGCANCEL

To perform HMDE operations using FCU without the GUI (see Figures B.1, B.2, and B.3):

1. If you will be using an FCU parameter definition file to perform HMDE operations, make sure that the file contains the correct FCU initiation parameter sets for the HMDE operations you want to perform. If you will not be using the default FCU parameter definition file, note the name and location of the file.

2. Log in as root on the UNIX server, and enter: **fcunw [-nc] [param]**
   For example (see Figures B.1, B.2, and B.3):
   - To perform the HMDE operations in the default FCU parameter definition file with confirmations, enter: **fcunw**
   - To perform the HMDE operations in the default FCU parameter definition file without confirmations, enter: **fcunw -nc**
   - To perform the HMDE operations in a different FCU parameter definition file with confirmations, enter: **fcunw /directory/directory/file_name**
   - To perform the HMDE operations in a different FCU parameter definition file without confirmations, enter: **fcunw -nc /directory/directory/file_name**
   - To perform one specific HMDE operation with confirmations, enter:
     fcunw -P mto VSN:dataset targetfile No No No
   - To perform one specific HMDE operation without confirmations, enter:
     fcunw -nc -P mto VSN:dataset targetfile No No No

3. If you specified the **-nc** option, FCU will perform all specified HMDE operations without requesting confirmation for the FCU parameters or for existing HMDEmto target files.
   - If you did not specify the **-nc** option, FCU will display the FCU initiation parameters for the operation to be performed and request confirmation. Enter **ok** to perform the specified HMDE operation, or enter **cancel** to load the next set of FCU parameters.
   - If you did not specify the **-nc** option and the HMDEmto target file already exists, FCU will request confirmation to overwrite the file. Enter **ok** to overwrite the existing file, or enter **cancel** to load the next set of FCU initiation parameters.

4. When the HMDE operation starts, FCU displays **Start**. When the operation completes successfully, FCU displays **Complete**. If the operation does not start or complete successfully, FCU displays an error message and loads the next parameter set.

5. When the last FCU initiation parameter set is processed (or canceled by the user), the FCU program terminates and returns an ending status value. The ending status is included in **$status** for C-shell and **$?** for B-shell/K-shell.

   0 = successful completion. All HMDE operations completed successfully.
   1 = unsuccessful completion. One or more operations did not complete successfully.

```
# fcunw                                                    ← Start FCU with checking.
File Conversion Utility Ver.01-01-40/00                    ← FCU program version.

mto VSN:dataset file_name EA No LF                         ← First set of parameters.
ok/cancel ? ok                                             ← Enter ok or cancel.
Now checking...                                            ← Checking for target file.
Complete                                                   ← Operation completed.

otm file_name VSN:dataset EA No No                         ← Next set of parameters.
ok/cancel ? ok                                             ← Enter ok or cancel.
Input file : Open error (-350)                             ! Error info displayed.
  (Fal error    : xxx
  (System error : xxx

mto VSN:dataset file_name EA No LF                         ! Next set of parameters.
ok/cancel ? ok                                             ! Enter ok or cancel.
Now checking...                                            ! Checking for target file.
OverWrite ? ok/cancel ? ok                                 ! Enter ok to overwrite file.
Complete                                                   ! Operation completed.

mto VSN:dataset file_name EA No LF                         ! Next set of parameters.
ok/cancel ? cancel                                         ← Enter ok or cancel.
 :
 :
#
```

Figure B.1 Using FCU From the UNIX Command Line

```
# fcunw -nc                                                    ! Start FCU without checking.
File Conversion Utility Ver.01-01-40/00                        ! FCU program version.

mto VSN:dataset file_name EA No LF                             ! First set of parameters.
Now checking...                                                ! Starting HMDE operation.
Complete                                                       ! Operation completed.

otm file_name VSN:dataset EA No No                             ! Next set of parameters.
Input file : Open error (-350)                                 ! Error info. displayed.
  (Fal error    : xxx
  (System error : xxx

mto VSN:dataset file_name EA No LF                             ! Next set of parameters.
Now checking...                                                ! Starting HMDE operation.
Complete                                                       ! Operation completed.

mto VSN:dataset file_name EA No LF                             ! Next set of parameters.
 :
 :
#
```

Figure B.2 Using the -nc Option

```
# fcunw -nc -P mto VSN:dataset file_name EA No LF              ! Start FCU without checking.

mto VSN:dataset file_name EA No LF                             ! Specified FCU parameters.
Now checking...                                                ! Starting HMDE operation.
Complete                                                       ! Operation completed.
#
```

Figure B.3 Using the -P param Option

B.2 Listvol VSN Function

The **listvol VSN** function enables FCU users to access the S/390 dataset information without having to launch the FCU GUI (and use the **Help-MF-File** command). The **listvol VSN** UNIX command displays the dataset information for the specified VSN (see Figure B.4). The **listvol VSN** function requires the HMDE volume definition file (see section 4.1).

Note: The **listvol VSN** function is available in FCU for UNIX versions 01-01-41 and later.

```
# listvol volser                                              ! Specify 6-character VSN.

  Dataset Name      DO     RF    RL     BL     TT    R     EX (Cyl:Trk)
_____________________________________________________________________
*SAMFILE01.FIX      SAM    F     4096   4096   1     10    100:0
-DAMFILE.F          DAM    F     128    4096   0     10    100:0
?SAMFILE.VSE        SAM    ?     0      0      0     0     0:0

0                                                           ! Return value (normal end).
#
```

Figure B.4 Listvol VSN Function

The **listvol VSN** function displays the following information:

- **Dataset name**. An asterisk (*) before the dataset name indicates that FAL/FCU can process the dataset. A dash (-) indicates that FAL/FCU cannot process the dataset. A question mark (?) before the dataset name indicates that FCU can process the dataset only if the VSE record option is used to specify the RF, RL, and BL.
- Dataset organization (**DO**) type: SAM, DAM, PAM, VSAM, ??? = unknown. FAL/FCU supports SAM datasets.
- Record format (**RF**): F = fixed length, V = variable length, U = undefined length, S = spanned record, ? = unknown. FAL/FCU supports F and V record formats.
- Record length (**RL**): in bytes
- Block length (**BL**): in bytes
- **TT**+**R**: last block address
- **EX (Cyl:Trk)**: data extent size (number of cylinders:number of tracks)
- Return value: **0** indicates normal end; **1** indicates error end. If an error occurred, the error code and message are displayed and the error is logged in the error log file.

Appendix C FAL/FCU Error Codes

C.1 FAL Error Codes

The error information returned by the **datasetGetLastError** function includes the FAL error information defined in the **dataset.h** file. Table C.1 lists and describes the FAL error codes and provides instructions for resolving each error condition. In Table C.1, the error codes marked by an asterisk (*) may also be reported when I/O access contention occurs. If the cause of the error cannot be identified as described in Table C.1, check for illegal I/O access contention for the HMDE volume between the S/390 and open-system hosts.

The FAL error logs for UNIX are **/tmp/fal_error** and **/tmp/fal_error.bak**, and **/tmp/fal_dump** and **/tmp/fal_dump.bak**. The FAL logs for Windows NT are **c:\fal_error** and **c:\fal_error.bak**, and **c:\fal_dump** and **c:\fal_dump.bak**.

Note: Error codes with a negative value are FAL errors. Error codes with a positive value are system errors. UNIX system error codes are defined in the standard error file **errno.h**.

Table C.1 FAL Error Codes

Error Code	Error Message and Description	Recommended Action(s)
-7*	DATASET_ERROR_INVALID_VOLUME The actual VSN and the VSN specified in the HMDE volume definition file do not match.	Make sure that the VSN in the HMDE volume definition file is correct.
-8	DATASET_ERROR_DATASET_NOT_FOUND The target dataset was not found.	Make sure that the actual dataset name and the specified dataset name are the same. You can use the MF-File list command in the FCU HELP menu, or VTOC dump data on the S/390 host, to check the dataset name.
-9	DATASET_ERROR_NOT_SUPPORTED The data format is not supported.	Make sure that the dataset was created correctly on the S/390 host.
-10*	DATASET_ERROR_DEVICE_TYPE_NOT_SUPPORTED The device emulation type is not supported.	Make sure that the device emulation type (LVI) is correct in the HMDE volume definition file. The supported LVIs are 3390-3A, -3B and -3C; and 3380-KA, -KB, and -KC.
-11	DATASET_ERROR_DSORG_NOT_SUPPORTED The dataset organization type is not supported.	Check the DO type using the MF-File list command in the FCU HELP menu, or VTOC dump data on the S/390 host.
-12	DATASET_ERROR_RECFM_NOT_SUPPORTED The record format is not supported.	Check the RF type using the MF-File list command in the FCU HELP menu, or VTOC dump data on the S/390 host.
-13*	DATASET_ERROR_INVALID_DATA The data in the VTOC or the dataset is invalid.	Make sure that the VTOC and dataset were created correctly on the S/390 host.
-14*	DATASET_ERROR_VOLUME_DEFINITION_INVALID The format of volume definition file is invalid.	Make sure that the HMDE volume definition file was created correctly.
-15	DATASET_ERROR_DATASET_NOT_OPENED An attempt was made to read the dataset without opening it.	Make sure that the **datasetOpen** function is called before the **datasetGet** function.

Table C.1 FAL Error Codes (continued)

Error Code	Error Message and Description	Recommended Action(s)
-16	DATASET_ERROR_DATASET_NOT_CLOSED An attempt was made to open the dataset without closing it first.	Make sure the requirements and restrictions specified in Chapter 6 are met, for example: Dataset open and close must be used as a pair. More than one dataset cannot be open within one process. **datasetOpen**, **datasetGetFileInformation**, and **datasetFindFirstFile** cannot be used while the dataset is being accessed by **datasetGetFileInformation** or **datasetFindFirstFile**. **datasetGetFileInformation** and **datasetFindFirstFile** cannot be used while the dataset is being accessed **datasetOpen**.
-17	DATASET_ERROR_BUFLEN_SHORT The buffer length specified by **datasetGet** is shorter than the actual record length.	Make sure that the buffer area is larger than the dataset record length.
-18*	DATASET_ERROR_VOLUME_LABEL_INVALID No standard volume label was found, or the contents of the VTOC are illegal.	Make sure that volume initialization is complete and correct on the S/390 host.
-19*	DATASET_ERROR_VTOC_INVALID No VTOC found, or contents of VTOC are invalid.	Make sure that the VTOC was created correctly on the S/390 host.
-20*	DATASET_ERROR_VOLUME_NOT_DEFINED The specified volume is not defined.	Make sure that the specified volume has been entered correctly in the HMDE volume definition file.
-21	DATASET_ERROR_INVALID_ARGUMENT An argument of the function is invalid.	Make sure that the argument for the FAL function is correct.
-22	DATASET_ERROR_NO_DATASET No dataset was found.	Make sure that the dataset has been created correctly on the S/390 host.
-23*	DATASET_ERROR_NON_STANDARD_R0_EXIST Nonstandard record 0 (R0) exists.	Change the R0 track format to standard track format. FAL cannot write on tracks with nonstandard R0.
-24	DATASET_ERROR_INVALID_MODE The **mode** argument of **datasetOpen** is not valid.	Make sure that the value of the **mode** argument for the **datasetOpen** function is either **r** (for read) or **w** (for write).
-25*	DATASET_ERROR_VOLUME_DEFINED_READ_ONLY The open-system host tried to write to a read-only volume.	Make sure that the target dataset for an open-system write operation is on a 3390-3A/C or 3380-KA/C volume.
-27*	DATASET_ERROR_END_OF_VOLUME The end of volume was detected before the end of dataset was detected.	The open-system volume/partition size is smaller than the S/390 volume size. Make sure that the partition size is specified correctly on the open system.
-28	DATASET_ERROR_OVERFLOW Data cannot be written because the dataset is full.	Check the size of the data to be written, and extend the size of the dataset as needed.
-33	DATASET_ERROR_PARAMETER_MISMATCH User-specified RF, BL, RL does not match the RF, BL, RL defined in the VTOC; or RF, BL, RL not specified and not defined in VTOC.	Make sure to specify the correct VSE record option parameters when accessing VSE datasets (see section 3.1.6).
-35	DATASET_ERROR_NO_LICENSE FAL can't permit execution of software that doesn't have a software license.	Ensure that the software license is current and correct. If problems persist, please contact the Hitachi Data Systems Support Center..

-36	DATASET_ERROR_TIMEOUT_LICENSE FAL can't permit execution of software with an expired software license trail time.	Ensure that the trial software license is current and correct. If problems persist, please contact the Hitachi Data Systems Support Center
-37	DATASET_ERROR_HOSTNAME_CHANGE FAL can't permit execution if the current host and the installed host are not identical and/or the hostname is changed.	Ensure that the current host name has not been changed.
-50*	DATASET_ERROR_END_OF_FILE End of File (EOF) was detected.	None.
-51*	DATASET_ERROR_END_OF_VTOC End of VTOC was detected.	None.

C.2 FCU Error Codes for UNIX

If FCU for UNIX reports an error, use the **Help-Error** command to view the most recent error. Table C.2 lists and describes the FCU error codes for UNIX and provides instructions for resolving each error condition. In Table C.2, the error codes marked by an asterisk (*) may also be reported when I/O access contention for the HMDE volume occurs between the S/390 and open-system hosts. If the cause of the error cannot be identified as described in Table C.2, check for any illegal I/O contention for the HMDE volume.

Note: Error codes with a negative value are FCU errors. Error codes with a positive value are system errors. UNIX system error codes are defined in the standard error file **errno.h**.

Table C.2 FCU Error Codes for UNIX

Error Code	Error Message and Description	Recommended Action(s)
-100	No parameter file The FCU parameter definition file could not be found.	If you specified the parameter definition file using the [**param**] option, make sure that the specified file exists and the name is correct. If you did not specify the [**param**] option when you started FCU, make sure that the default parameter definition file exists (**fcudata.param** in the current directory).
-101*	Parameter file: Open error An error occurred when opening the parameter definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-102*	Parameter file: Read error An error occurred when reading the parameter definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-103*	Parameter file: No valid data The parameters in the parameter definition file are not valid.	Make sure that the FCU initiation parameters are entered correctly in the parameter definition file.
-107	Parameter file: CODE_CONV error The code conversion specified in the parameter definition file is not valid.	Make sure that the code conversion is specified as either **EA** or **No**.
-108	Parameter file: PADDING error The padding option specified in the parameter definition file is not valid.	Make sure that the padding is specified as either **Yes** or **No**.
-109	Parameter file: DELIMITER error The delimiter option specified in the parameter definition file is not valid.	Make sure that the delimiter is specified as either **CR**, **LF**, or **No**.
-110*	Parameter file: Open error An error occurred when opening and outputting the parameter definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-111	Parameter file: Write error An error occurred when writing to the parameter definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-112	Parameter file: Close error An error occurred when closing the parameter definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-114	Parameter: No input file name The input file name was not specified.	Make sure to specify the input file name.
-115	Parameter: VSN error The specified VSN is not correct.	Make sure that the specified VSN matches the actual VSN. Make sure that the VSN is separated from the dataset name by a colon (:).
-116	Parameter: Input file name error The specified input file name is not correct.	Make sure that the specified file name matches the actual file name.
-117	Parameter: Dataset name error The specified input dataset name is not correct.	Make sure that the specified dataset name matches the actual dataset name.
-118	Parameter: Output file name error The specified output file name is incorrect.	Make sure that the specified output file name matches the actual output file name.
-119*	Input file: Open error An error occurred when acquiring the dataset attribute information of the input file.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual. For example, if a partition name does not match the partition name in the volume definition file, system error code 6 (No such device) is displayed.
-120	Overwrite ? (OK/Cancel) This message asks you to confirm whether to overwrite the existing file.	The specified open-system target file already exists. Select **OK** to overwrite the file, or select **Cancel** to specify a different target file.
-121	Output file: File name error The output file name is not specified.	Make sure that the correct output file name is specified.
-122*	Output file: Open error An OPEN error occurred when checking to see if the output file exists.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-125*	Volume definition: VSN error The VSN specified in the volume definition file is incorrect.	Display the contents of the volume definition file using the **Help-Volume** command. Make sure that the VSN for the specified volume is correct.
-126	Volume definition: Partition name error The partition name specified in the volume definition file is incorrect.	Display the contents of the volume definition file using the **Help-Volume** command. Make sure that the partition name is correct.
-127*	Volume definition: Emulation type error The LVI type specified in the volume definition file is incorrect.	Display the contents of the volume definition file using the **Help-Volume** command. Make sure that the LVI type is correct.
-128*	Volume definition file: Open error An error occurred when opening the volume definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors. For example, if the volume definition file does not yet exist, error code 2 (No such file or directory) is displayed.
-129*	Volume definition file: Read error An error occurred when reading the volume definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-130*	Volume definition file: No data The information found in the volume definition file is not valid.	Display the contents of the volume definition file using the **Help-Volume** command. Make sure that the parameters for each volume are correct.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-131	Volume definition file: Close error An error occurred when closing the volume definition file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-135	Parameter error: No input file name The input VSN is not specified.	Specify the VSN of the S/390 source dataset before selecting the **Help-MF-File** command.
-136	Parameter error: VSN error The input VSN is incorrect.	Make sure that the VSN has six characters.
-137	Dataset error: No dataset The specified volume has no datasets.	Make sure that the VSN is correct.
-138*	Dataset error: Search error An error occurred when searching the dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-139	Dataset error: Close error An error occurred when closing the dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-140	Input file error: Invalid organization type The DO type of the dataset is not supported.	Display the attribute information using the **Help-MF-File** command. The DO type must be SAM.
-141	Input file error: Invalid record format The RF type of the dataset is not supported.	Display the attribute information using the **Help-MF-File** command. The RF type must be fixed-length or variable-length.
-142	Input file error: Invalid block length The block length of the dataset is invalid.	Display the attribute information using the **Help-MF-File** command. The block length must be nonzero and cannot be greater than 32 kB.
-143	Input file error: Invalid record length The record length of the dataset is invalid.	Display the attribute information using the **Help-MF-File** command. The record length must be nonzero and cannot be greater than 32 kB.
-144*	Input file error: No data No data was found in the specified dataset.	Display the attribute information using the **Help-MF-File** command, and check the dataset size.
-150*	Input file: Open error A file open error occurred in the input dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-151*	Output file: Open error A file open error occurred in the output UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-152	Output file: Get file data error A data acquisition error of the output file occurred during an HMDEotm operation.	Collect information such as error log for troubleshooting.
-153	Processing data: Length check error A data length to be processed by HMDEotm does not match.	Make sure that the specified data length matches the actual data length. Collect information such as error log for troubleshooting.
-155	Buffer: Memory allocation error Memory allocation failed.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-160*	Input file: Read error A read error occurred in the input dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-161*	Output file: Write error A write error occurred in the output UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-162	Output file: Code conversion error An error occurred in the code conversion to the output UNIX file.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1.
-163	Get processing data error The acquisition of processing data failed.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-165	Dataset error: Invalid data An invalid record length was found in the dataset.	Make sure that the S/390 dataset was generated correctly.
-170	Input file: Close error A file close error occurred in the input dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-171	Output file: Close error A file close error occurred in the output UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-180	UNIX/Open system file: Invalid directory name The specified directory name is not valid.	Check the specified directory name.
-181	UNIX file: Not a directory The specified name is not a directory name.	Check the specified directory name.
-182*	UNIX/Open system file: Open directory error A directory open error occurred.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-183	UNIX/Open system file: Close directory error A directory close error occurred.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-190	Input file name: No data The input file name is not specified.	The input file name must be specified when you select **File-Save**.
-191	Output file name: No data The output file name is not specified.	The output file name must be specified when you select **File-Save**.
-192	Parameter file name: No data The parameter definition file name is not specified.	The FCU parameter definition file name must be specified when you select **File-Save**.
-200	Parameter file: End line The last parameter set was loaded from the parameter definition file.	The next time you select **File-Load**, the first set of parameters will be loaded.
-201	Parameter file: Direction error The data transfer direction specified in the parameter file is incorrect.	Make sure that the direction (**mto** or **otm**) is correct.
-202	Parameter file: Too many data The number of parameter sets for parameter definition file exceeded 100.	The FCU parameter definition file can only store a maximum of 100 parameter sets. If necessary, delete one or more parameter sets to make room for a new parameter set.
-203	Parameter: Empty select error The **Emp** parameter is incorrect.	Make sure that the **Emp=Yes/No** parameter is correct.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-204	Parameter: RDW select error The **RDW** parameter is incorrect.	Make sure that the **RDW=Yes/No** parameter is correct.
-205	RDW error: CODE_CONV not supported Code conversion is not specified as **No** when **RDW=Yes**.	Code conversion cannot be performed when **RDW=Yes**. Change the code conversion parameter to **No**.
-206	RDW error: PADDING not supported Padding is not specified as **No** when **RDW=Yes**.	Padding cannot be processed when **RDW=Yes**. Change the padding parameter to **No**.
-207	RDW error: DELIMITER not supported Delimiter is not specified as **No** when **RDW=Yes**.	Delimiters cannot be processed when **RDW=Yes**. Change the delimiter parameter to **No**.
-210	Parameter file: Comment line This is a comment line in the parameter file.	If you specify **Load**, FCU will move to the next line. You can also replace the comment line with a valid parameter.
-220	Parameter: VSE select error The VSE parameter format is not correct.	Make sure that the number of VSE parameters is correct and that a comma is used correctly to separate the VSE parameters.
-221	Parameter: VSE record format error Record format in the VSE parameter is not correct.	Make sure that the record format is set to either one of F/FB/V/VB.
-222	Parameter: VSE record length error Record length in the VSE parameter is not correct.	Make sure that the record length is set to the correct value within the extent allowed.
-223	Parameter: VSE block length error Block length in the VSE parameter is not correct.	Make sure that the block length is set to the correct value within the extent allowed.
-230	No code conv. table file: No code conv. table The code conversion table was not found.	Make sure that the code conversion table file name is correct and that the file exists. This error may also be reported if you mix mainframe and OPEN-3 devices in the same HMDE volume definition file.
-231	Code conv. table: Open error The code conversion table could not be opened.	Refer to the OS user manuals for assistance.
-233	Code conv. table: Close error The code conversion table could not be closed.	Refer to the OS user manuals for assistance.
-234	Code conv. table: Get file data error The size of the code conversion table could not be obtained.	Check the contents of the file. Refer to the OS user manuals for assistance.
-235	Code conv. table: File size error The size of the code conv table is not correct.	Make sure that the size of the code conversion table is 256 bytes.
-236	Code conv. table function: Invalid argument No source data to be converted was found.	Check the contents of the input file, especially the delimiters.
-238	Code conv. table name: No data The file name of the code conversion table is not specified.	If you do not specify **EA** or **No** for the code conversion option, make sure to specify the correct file name of your code conversion table.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-300	Data error: Invalid record length The data length is not correct for the HMDEotm padding function.	Check the source data length and the target record length, and make sure that the record length is correct for the source data entities.
-301	Dataset error: Invalid record format The record format is not correct for the HMDEotm padding function.	For HMDEotm with padding, make sure that the target dataset has fixed-length record format.
-302	Parameter error: Delimiter error The delimiter setting is not correct for the HMDEotm padding function.	If padding=Yes for an HMDEotm operation, the delimiter option must be CR, LF or CRLF.
-319*	Dataset: Open error An error occurred when opening the dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual. For example, if the partition name does not match the partition name in the volume definition file, system error code 6 (No such device) is displayed.
-324	O to M error: RDW is not supported	Do not specify the RDW option for HMDEotm operations.
-340	Dataset error: Invalid organization type The DO type of the dataset is not supported.	Display the attribute information using the **Help-MF-File** command. The DO type must be SAM.
-341	Dataset error: Invalid record format The RF of the dataset is not supported.	Display the attribute information using the **Help-MF-File** command. The RF type must be fixed-length or variable-length.
-342	Dataset error: Invalid block length The block length of the dataset is invalid.	Display the attribute information using the **Help-MF-File** command. The block length must be nonzero and cannot be greater than 32 kB.
-343	Input file error: Invalid record length The record length of the dataset is invalid.	Display the attribute information using the **Help-MF-File** command. The record length must be nonzero and cannot be greater than 32 kB.
-350*	Input file: Open error An open error occurred in the input UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-351*	Output file: Open error A file open error occurred in the output dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-352	Input file: Get file data error A data acquisition error for input file occurred during an HMDEotm operation.	Collect information such as error log for trouble shooting.
-353	Processing data: Length check error A data length to be processed in HMDEotm operation does not match.	Collect information such as error log for trouble shooting.
-355	Buffer: Memory allocation error Memory allocation failed.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-360*	Input file: Read error A read error occurred in the input UNIX file.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-361*	Output file: Write error A write error occurred in the output dataset.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-362	Output file: Code conversion error An error occurred in the code conversion to the output dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1.

Table C.2 FCU Error Codes for UNIX (continued)

Error Code	Error Message and Description	Recommended Action(s)
-363	Get processing data error The acquisition of processing data failed.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-370	Input file: Close error A file close error occurred in the input UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-371	Output file: Close error A file close error occurred in the output dataset.	Display the error code using the **Help-Error** command. If a FAL error code is displayed, refer to Table C.1. If a system error code is displayed, please refer to the OS user manual.
-379*	UNIX file: No data No data was found in the input UNIX file.	Make sure to specify an input file which contains data.
-380	No UNIX file The specified UNIX file was not found.	Make sure that the specified UNIX file exists.
-381*	UNIX file: Open error An open error occurred in the UNIX file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-382	Output file: Unsupported record format The record format of the output file is not supported.	Display the attribute information using the **Help-MF-File** command. The RF type must be fixed-length or variable-length.
-383*	Input file: Invalid format The format of the input file is incorrect.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-384*	Input file: Invalid delimiter position The delimiter position in the input file is incorrect. Data record length of input file exceeds that of target dataset, or a record with no data entity is included.	Display the attribute information using the **Help-MF-File** command. Make sure that the record length of the target dataset is correct.
-385	Input file: File seeking error An error occurred when seeking for the input file.	Display the system error code using the **Help-Error** command. Please refer to the OS user manual for information on system errors.
-400	Parameter: Invalid input file name More than one input file name was specified.	Specify only one file name as the input file.

C.3 FCU Error Codes for Windows NT

If FCU for Windows NT reports an error, use the **View-Error information…** command to view the most recent error. FCU for Windows NT also logs errors in the FCU log file (e.g., **fcudata.prm.log**). Table C.3 lists and describes the FCU error codes for Windows NT and provides instructions for resolving each error condition.

Note: Error codes with a plus value are system errors. Windows NT system error codes are defined in the **errno.h** file attached with Microsoft Visual C++.

Table C.3 FCU Error Codes for Windows NT

Error Code	Error Message and Description	Recommended Action(s)
-100	Parameter definition file: Open error An error occurred when opening the parameter definition file.	Make sure that the parameter definition file was created correctly. If the parameter definition file was created correctly, check the system error.
-101	Parameter: Count error An error is detected in the parameter count.	Make sure that the parameter count is correct.
-102	Parameter: Direction error The data transfer direction is not correct.	Make sure that the direction is specified correctly as **mto** or **otm**.
-103	Parameter: Mainframe file name error Mainframe file name is not correct.	Make sure that the mainframe file name is set correctly.
-104	Parameter: Open system file name error Open system file name is not correct.	Make sure that the open system file name is set correctly.
-105	Parameter: Code conversion error Code conversion setting is not correct.	Make sure that the code conversion option is specified as **EA**, **EcA**, **No**, or **File_name** (of your code conversion table). This error may also be reported if you mix 3390/3380 and OPEN-3 devices in the same HMDE volume definition file.
-106	Parameter: Padding error Padding setting is not correct.	Make sure that the padding option is specified as **Yes** or **No**.
-107	Parameter: Delimiter error Delimiter setting is not correct.	Make sure that the delimiter option is specified as **CRLF** or **No**.
-108	Parameter: Add parameter error Delimiter setting is not correct.	If you are adding delimiters for Windows NT, make sure that the delimiter option is specified as **CRLF** (not just **CR** or **LF**).
-109	Parameter: Empty duplication error More than one empty setting is specified.	Specify only one empty setting.
-110	Parameter: RDW duplication error More than one RDW setting is specified.	Specify only one RDW setting.
-120	Volume definition file: Open error An error is detected when opening the volume definition file.	Make sure that the volume definition file was created correctly. If the volume definition file is correct, check the system error.
-130	Dataset: No dataset error No dataset is found.	Make sure that the mainframe name is specified correctly, or that the dataset is allocated correctly on the specified volume.
-131	Dataset: Search error An error is detected in searching the dataset.	Make sure that the volume definition file name is specified correctly, or that the mainframe file name is specified correctly.

Table C.3 FCU Error Codes for Windows NT (continued)

Error Code	Error Message and Description	Recommended Action(s)
-132	Dataset: Information get error An error is detected in acquiring dataset information.	Make sure that the volume definition file name is specified correctly, or that the mainframe file name is specified correctly.
-133	Dataset: Organization error The specified dataset org. type is not correct.	Make sure that the dataset organization type is specified correctly.
-134	Dataset: Record format error The specified record format is not correct.	Make sure that the record format is specified correctly.
-135	Dataset: Block length error The specified block length is not correct.	Make sure that the block length is specified correctly.
-136	Dataset: Record length error The specified record length is not correct.	Make sure that the record length is specified correctly.
-137	Dataset: Dataset size error The specified dataset size is not correct.	Make sure that the dataset size is specified correctly.
-138	Dataset: Close error An error is detected during close operation.	Check the FAL error code and system error code.
-150	Mainframe file: Open error An error is detected when opening the mainframe file.	Check the FAL error code and system error code.
-151	Mainframe file: Read error An error is detected during reading data from the mainframe file.	Check the FAL error code and system error code.
-152	Mainframe file: Write error An error is detected when writing data into the mainframe file.	Check the FAL error code and system error code.
-153	Mainframe file: Close error An error is detected when closing the mainframe file.	Check the FAL error code and system error code.
-154	Mainframe file: Record format error An error is detected in the record format of the mainframe file.	For HMDEotm with the padding function, make sure that the target dataset has fixed-length record format (or change padding to **No**).
-170	Open system file: Open error An error is detected when opening the open system file.	Make sure that the open-system file name is specified correctly. Check if any system error is reported.
-171	Open system file: Read error An error is detected when reading data from the open system file.	Check the system error.
-172	Open system file: Write error An error is detected when writing data into the open system file.	Check the system error.
-173	Open system file: Close error An error is detected when closing the open system file.	Check the system error.
-174	Open system file: No data error No dataset is found.	Make sure that the open-system file has data. If not, create the appropriate data in the open-system file.

Table C.3 FCU Error Codes for Windows NT (continued)

Error Code	Error Message and Description	Recommended Action(s)
-175	Open system file: Delimiter (CR) position error Delimiter (CR) position error is detected. The source data record length exceeds the target record length, or a record with no data entity is included.	Make sure that the open-system file name is correct. Make sure that the mainframe dataset name is correct. Make sure that the record length of the open-system file is correct.
-176	Open system file: Delimiter (LF) position error Delimiter (LF) position error is detected.	Make sure that the open-system file name is correct. Make sure that the mainframe dataset name is correct. Make sure that the record length of the open-system file is correct.
-177	Open system file: Record format error An illegal record format is found.	Make sure that the open-system file name is correct. Make sure that the mainframe dataset name is correct. Make sure the record format (fixed- or variable-length) of the open-system file data is correct.
-178	Open system file: Record length error An illegal record length was found. Data length of open-system file is too large.	Check the data length of the open-system file, and make sure the dataset has the correct record length.
-190	Code conversion error An error was found during code conversion.	Make sure that the dataset size is specified correctly.
-200	Process data get error An error is detected during close operation.	Check the FAL error code and system error code.
-220	External table file: Open error The code conversion table could not be opened.	Check the file name of code conversion table. Check the system error.
-221	External table file: Size error The code conversion table size is not correct.	Make sure that the size is 256 bytes and that the table was created correctly.
-222	External table file: Read error A read error was found when reading the code conversion table.	Check the system error.
-223	External table file: Close error The code conv. table could not be closed.	Check the system error.
-240	Parameter: Direction, PAD, and DEL not matched The combination of **otm** direction, **PAD=Yes**, and **DEL=No** is not allowed.	For HMDEotm with the padding function, make sure that the delimiter option is specified as **Yes** (or set padding=No).
-241	Parameter: Direction and RDW not matched The combination of **otm** data transfer direction and **RDW=Yes** is not allowed.	When the HMDE data transfer direction is **otm**, make sure that the RDW option is specified as **No**.
-242	Parameter: Code conv. and RDW not matched The combination of **RDW=Yes** and code conversion other than **No** is not allowed.	When the code conversion option is **EA** or **File_name**, make sure that the RDW option is specified as **No**. When RDW=Yes, the code conversion option must be specified as **No**.
-243	Parameter: Padding and RDW not matched The combination of **RDW=Yes** and **padding=Yes** is not allowed.	When the padding option is specified as **Yes**, make sure that the RDW option is specified as **No**. When the RDW option is specified as **Yes**, make sure that the padding option is specified as **No**.
-244	Parameter: Delimiter and RDW not matched The combination of **RDW=Yes** and **delimiter=Yes** is not allowed.	When the delimiter option is specified as **Yes**, make sure that the RDW option is specified as **No**. When the RDW option is specified as **Yes**, make sure that the delimiter option is specified as **No**.

Appendix D EBCDIC-ASCII Code Conversion

Table D.1 lists the EBCDIC-ASCII code conversions performed by the default code conversion table which is provided with FCU (refer to Table 3.1).

Table D.1 Default FCU EBCDIC-ASCII Conversions

Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII
00	NUL	NUL	20	DS		40	SP	DS	60	-	ENQ
01	SOH	SOH	21	SOS	a	41			61	/	BEL
02	STX	STX	22	FS	b	42			62		
03	ETX	ETX	23		c	43		s	63		
04	PF		24	BYP	d	44		t	64		
05	HT	RLF	25	LF	SMM	45		u	65		
06	LC	f	26	ETB	IL	46		v	66		
07	DEL	“	27	ESC	CUI	47		w	67		
08	GE	p	28		h	48		x	68		
09	RLF		29		i	49		y	69		
0A	SMM		2A	SW		4A		N	6A		
0B	VT	VT	2B	CUI		4B	.	ACK	6B	,	
0C	FF	FF	2C			4C	<	DC4	6C	%	LF
0D	CR	CR	2D	ENQ	HT	4D	(		6D	_	~
0E	SO	SO	2E	ACK	LC	4E	+	CUI	6E	>	
0F	SI	SI	2F	BEL	DEL	4F	|	@	6F	?	SUB
10	DLE	DLE	30			50	&	ETB	70		
11	DC1	DC1	31		j	51		z	71		
12	DC2	DC2	32	SYN	BS	52			72		
13	DC3	DC3	33		l	53			73		]
14	TM		34	PN	m	54			74		
15		e	35	RS	n	55		[	75		
16	BS	GE	36	UC	o	56			76		{
17	IL	g	37	EOT	PF	57			77		A
18	CAN	CAN	38		q	58			78		B
19	EM	EM	39		r	59			79	‘	-
1A	CC	k	3A		^	5A	!	SOS	7A	:	
1B	CUI		3B	CU3		5B	$	BYP	7B	#	
1C	IFS	IFS	3C	DC4	TM	5C	*	SW	7C	@	SP
1D	IGS	IGS	3D	NAK		5D	)		7D	‘	ESC
1E	IRS	IRS	3E			5E	:	CU3	7E	=	NAK
1F	IUS	IUS	3F	SUB	CC	5F	~	=	7F	“	FS

Table D.1 Default FCU EBCDIC-ASCII Conversions (continued)

Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII	Hex	EBCDIC	ASCII
80		C	A0		J	C0	{	#	E0	\	*
81	a	/	A1		V	C1	A		E1		
82	b		A2	s		C2	B		E2	S	
83	c		A3	t		C3	C		E3	T	
84	d		A4	u		C4	D		E4	U	
85	e		A5	v		C5	E		E5	V	
86	f		A6	w		C6	F		E6	W	
87	g		A7	x		C7	G		E7	X	
88	h		A8	y	‘	C8	H		E8	Y	
89	i		A9	z	:	C9	I		E9	Z	!
8A		D	AA		K	CA		Y	EA		4
8B		E	AB		L	CB		Z	EB		5
8C		F	AC		M	CC			EC		6
8D		G	AD	[	$	CD			ED		7
8E		H	AE		O	CE			EE		8
8F		I	AF		P	CF			EF		9
90			B0		Q	D0	}	‘	F0	0	
91	j		B1		R	D1	J		F1	1	
92	k	,	B2			D2	K	.	F2	2	SYN
93	l	%	B3			D3	L	<	F3	3	
94	m	_	B4			D4	M	(	F4	4	PN
95	n	>	B5			D5	N	+	F5	5	RS
96	o	?	B6			D6	O	|	F6	6	UC
97	p		B7			D7	P	&	F7	7	EOT
98	q		B8		\	D8	Q		F8	8	
99	r		B9			D9	R		F9	9	
9A	^	;	BA		S	DA			FA		
9B			BB		T	DB			FB		
9C			BC		U	DC		0	FC		
9D			BD	]	)	DD		1	FD		
9E			BE		W	DE		2	FE		
9F		}	BF		X	DF		3	FF		

Index