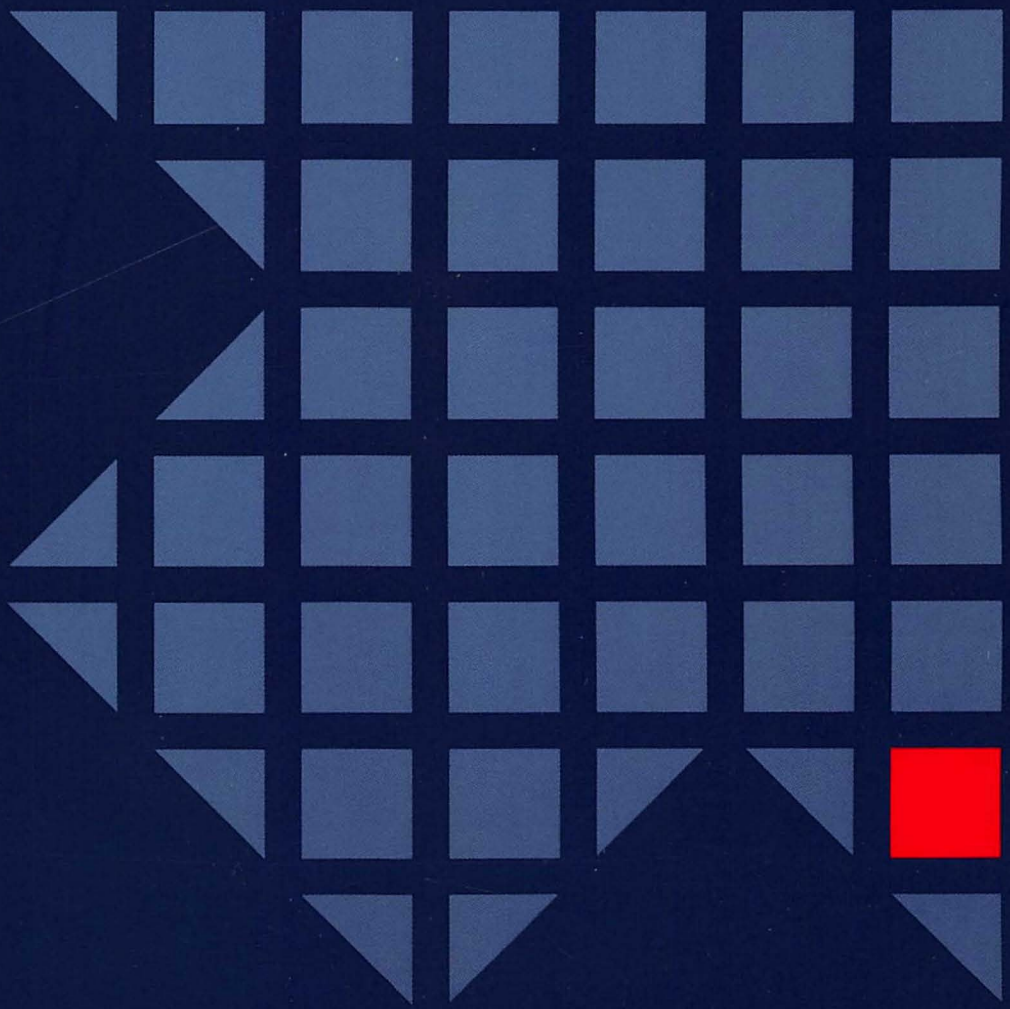
IBM

Virtual Machine/Extended Architecture
System Product

SC23-0364-3

Installation and Service

VM/XA SP Release 2.1



Virtual Machine/Extended Architecture System Product

SC23-0364-3

Installation and Service

VM/XA SP Release 2.1

Publishing Information

This publication was produced using BookMaster (Program Number 5668-015), BrowseMaster (Program Number 5688-009), the Document Composition Facility (Program Number 5748-XX9), and the composed Document Printing Facility (Program Number 5668-997).

All graphical illustrations and text in this manual were merged electronically without cut-and-paste operations.

Masters were printed on the IBM 4250 Printer.

This is a major revision of SC23-0364-2. See the "Summary of Changes" on page 901 for a summary of the changes made to this manual.

This edition applies to Release 2 Modification 1 of the Virtual Machine/Extended Architecture System Product (VM/XA SP) Licensed Program 5664-308. Technical changes and additions to the text and illustrations of these books are indicated by a vertical bar to the left of each change. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370, 30xx, 4300, and 9370 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to International Business Machines, Department 52Q/MS 511, Neighborhood Road, Kingston, N.Y. 12401. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Fourth Edition (June 1990)

Contents

Special Notices

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

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, N.Y. 10577.

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States and/or other countries:

IBM
MVS/ESA
MVS/SP
MVS/XA
Processor Resource/Systems Manager
PR/SM
System/370
Virtual Machine/Extended Architecture
VM
VM/XA
3090

Preface

Purpose

This manual describes how to:

- Install the Virtual Machine/Extended Architecture System Product (VM/XA SP) on a 370-XA processor
- Apply corrective and preventive service to the VM/XA System Product Release 2.1.

This manual is intended to help you install and service VM/XA System Product Release 2.1. It primarily contains installation and service information. The information in this manual must not be used for programming purposes.

Superseded Information

This edition contains information from, and supersedes, the following documents:

- *VM/XA SP Installation and Service*, SC23-0364-2
- *VM/XA SP Serviceability Enhancements Program Update Information APAR VM37518*, GC23-0503-0
- *VM/XA SP Release 2.1 Installation and Service Technical Newsletter*, SN31-1592
- *VM Program Update Information Enhancements for Receiving Electronic Service*, GC24-0533-0
- The "Installation and Service" chapter of *VM/XA SP 2.1 Release Guide*, GC23-0510-0.

Prerequisite APARs

The following APARs are prerequisites to this edition:

VM39596
VM40518
VM40833
VM41018
VM41040
VM41097
VM41159.

Audience

This publication is intended for those people responsible for installing and applying service to the VM/XA System Product Release 2.1.

Prerequisite Knowledge

To get the most out of this book, you should have a general idea of what the VM/XA System Product Release 2.1 does and what a virtual machine is. You should also have a general understanding of System/370 and 370-XA data processing techniques.

How to Use This Publication

Whenever you need to perform a system task, this publication describes the task in one- and two-column format.

1. *Substeps* in any task will be numbered and appear in one-column format, just like the lines you are now reading.

Screen output and input will appear on the left side of a two-column format. An example of screen *output* is a system message:

```
DMSACC724I '195' REPLACES
  'A (191)'
```

Information about screen output and input appear on the right side of a two-column format. For example:

This message tells you that minidisk 195 is now the A-disk. The indentation of the second line indicates that this message appears all on one line on the display terminal.

Screen output may appear in all capital letters or in mixed-case. Variable information is always in italic small letters. For instance:

```
REPLY value TO THE PROMPT
   "CYL|BLK ADDRESS"
```

If screen output or input contains variable information, the information is explained on the right side. For example:

value is a cylinder/block address; it may be the number 68, 105, or 66.

Input that you must enter appears on the left side of the two-column format in blue lowercase bold letters. The square bullet (■) next to input lines indicates you must press the **ENTER** key on the display console. For instance:

```
access 191 a ■
```

The right side of the two-column format will also explain input that you must enter. For instance:

Issue this command to make your 191 minidisk the A-disk.

Sometimes input you must enter varies. In such cases, *italicized variables* will appear. For instance:

```
input rdevno devtype xasres ■
```

When the input you enter is variable, notes on the right side will explain the variable information. For example:

The second control statement is the input control statement. Identify the device number (*rdevno*) and the tape device type (*devtype*) where the Starter System Tape is mounted.

Note: The output that is documented in this book may not exactly match what appears at the display terminal. However, important messages and prompts that are necessary to perform the system installation are included in the steps for installation.

Part 1 also uses figures to introduce an installation step and to aid you when you want to make quick references to installation steps. For example:

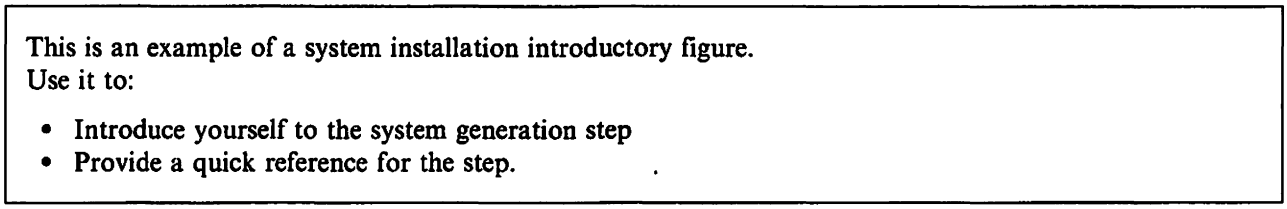


Figure 1. Example of System Installation Introductory Figure

Some Common Abbreviations and Variables

Here is a list of common abbreviations and variables used in this publication:

Abbreviation	Term
rdevno	Real device number
vdevno	Virtual device number
devtype	Device type
volid	Volume identification
userid, user ID	User identification for a virtual machine
fn	Filename
ft	Filetype
fm	Filemode
V=F	Virtual=Fixed
V=R	Virtual=Real
V=V	Virtual=Virtual

Related Publications

If you want to know the order number of a publication this book refers to, see "VM/XA System Product Publications" on page 909. This section also lists other hardware and software books that may help you perform virtual machine operation tasks.

You can find the titles of all VM/XA System Product Release 2.1 publications in Figure 69 on page 910.

Part 1. Installing the System

Part 1 of this book contains the following chapters:

- Chapter 1, "Introduction" on page 3, describes the starter system, the MAINT virtual machine, the difference between first level and second level installation, and planning considerations for installing the Group Control System (GCS) and for installing a new system national language.
- Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" on page 13, shows the process of installing VM/XA System Product Release 2.1 with the starter system at first level.
- Chapter 3, "Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)" on page 113, shows the process of installing VM/XA System Product Release 2.1 with the starter system at second level.
- Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215, shows the process of installing VM/XA System Product Release 2.1 using an existing VM/SP or VM/SP HPO system.
- Chapter 5, "Installing a New System National Language" on page 311, shows how to install a new default system national language.

Chapter 1. Introduction

Part 1 describes the step-by-step installation procedures for the control program (CP), conversational monitor system (CMS), dump viewing facility (DV), and group control system (GCS).

In Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)," and Chapter 3, "Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)," the installation steps require you to use a Virtual Machine/Extended Architecture* System Product (VM/XA* SP) starter system. The starter system is a functional VM/XA System Product system that you can use to tailor VM/XA System Product to your processor configuration. The starter system contains a full-screen editor that aids in creating and modifying system definition files—the system control file (HCPSYS ASSEMBLE), the real I/O configuration file (HCPRIO ASSEMBLE), the user directory (USER DIRECT), and the CMS generation file (DMSNGP ASSEMBLE).

To use the starter system, you must:

- Restore the system from the starter system tape that IBM* supplies
- Perform a hardware initial program load (IPL) from the system residence device (XASRES).

The starter system tape contains the DASD DUMP/RESTORE (DDRXA) program in the first file. DDRXA is a stand-alone program that restores the starter system from tape to the system residence device. Also included in the first file is the Device Support Facilities program; during installation, this program inspects DASD volumes being used for minidisks.

The starter system tape contains the VM/XA System Product starter system as the second file. The starter system is device dependent; that is, you must restore the system to its compatible device type. IBM supplies 3350, 3375, 3380, and 3390 starter systems.

Figure 2 shows the file arrangement of the starter system tape.

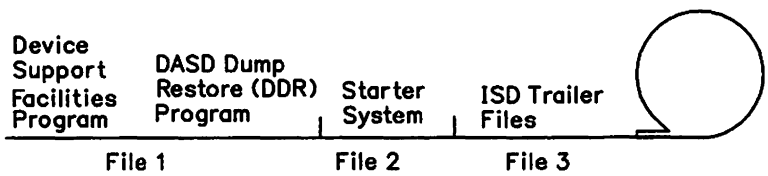


Figure 2. Starter System Tape

After you have installed the starter system, you have a vehicle to continue the VM/XA System Product installation. Now you must:

- Load onto disk the VM/XA System Product product tape that IBM supplies.

  Note: As soon as the CMS part of the product tape is loaded by the ITASK EXEC, CMS will be generated with IBM-supplied defaults. Later you can regenerate the CMS nucleus with your own defaults.

  Figure 3 shows the file arrangement of the product tape.

Virtual Machine/Extended Architecture, VM/XA and IBM are trademarks of the IBM Corporation.

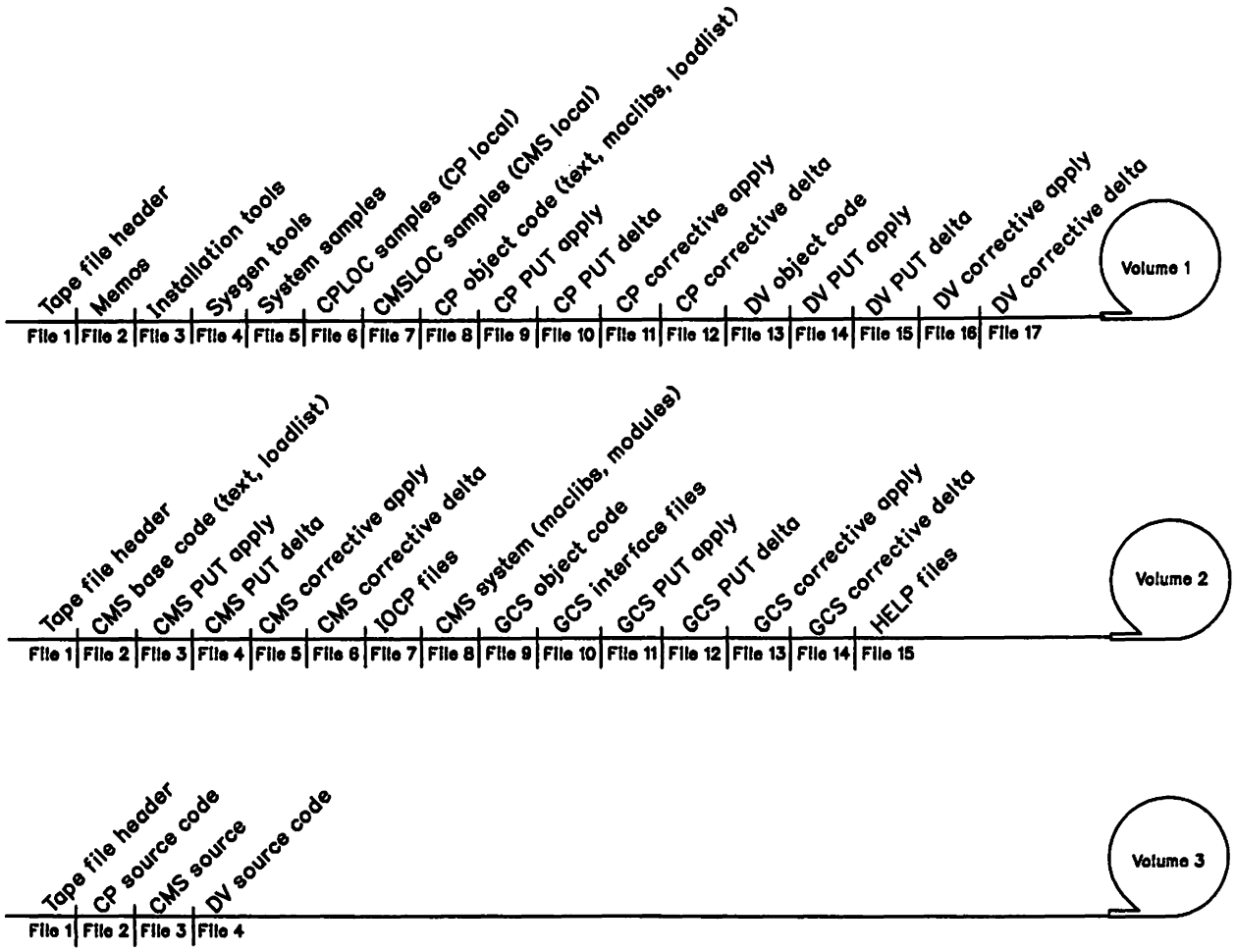


Figure 3. VM/XA SP Product Tape

Notes:

1. The term **product tape** refers to the whole set of three tape volumes.
2. The product tape is in VMSUP format. This means that PUT and COR service are shipped with the base product and are applied when you install the system.
3. The file arrangement has not changed for VM/XA System Product Release 2.1. Corrective service files, however, are now in the PUT apply and PUT delta tape files. The corrective apply and corrective delta files are dummies.

- Install Assembler H Version 2 Program Product.
- Edit IBM-supplied sample of CMS generation file (DMSNGP ASSEMBLE) then edit the IBM-supplied samples of the system control file (HCPSYS ASSEMBLE) and the real I/O configuration file (HCPRIO ASSEMBLE) or create your own system control and real I/O configuration files.
- Generate the conversational monitor system (CMS) and the control program (CP) components of VM/XA System Product.

  Note: You do not need to generate the dump viewing facility.
- Perform a hardware initial program load of the newly generated VM/XA System Product. IBM generates the dump viewing facility for you and it is loaded onto DASD from the product tape.
- Update the user directory (USER DIRECT).
- Install the Environmental Record and Editing Program.

- Make CMS into saved systems—a 370 system and a 370/XA system.
- Install the group control system.
- Create the CMSDOS, CMSBAM, CMSVSAM, CMSAMS, CMSINST, and HELP saved segments.

Special EXECs (ITASK EXEC, SPLOAD EXEC, and VMFBLD EXEC) do much of the system generation process. The EXECs operate interactively—this publication explains necessary responses and decisions you must make during the installation procedure.

The SPLOAD EXEC formats minidisks and loads product tape files to those minidisks.

The VMFBLD EXEC creates the CP, CMS, and GCS nuclei.

Note: In this publication, the term **system nucleus** means:

- A system that is resident in main storage,

  OR

- A loadable system that is resident on the system residence device.

MAINT Virtual Machine

Key to the system generation procedure is a virtual machine called MAINT. The MAINT virtual machine has a special minidisk configuration and an important set of command privilege classes; the minidisks and command classes allow MAINT to generate the VM/XA System Product system. Few virtual machines in the VM/XA System Product system have the same capabilities to alter the system.

The starter system has a user directory that defines the MAINT virtual machine for you. After you have initialized the starter system, you will use MAINT and continue with the system generation procedure.

Part of the generation procedure is loading the product tape. You will notice from screen messages that the product tape loads onto several of MAINT's minidisks. The reason for separating parts of the product tape onto several minidisks is that this strategy keeps the VM/XA System Product components separate; the strategy also separates source files and object files. The separations are important not only to the system generation procedure but also to IBM service. (For more information about IBM service, see Part 2, "Servicing the System" on page 347.)

If you want more information about the MAINT virtual machine, see "The MAINT Virtual Machine" on page 352.

System Installation on 308x, 4381, and 3090 Processor Complexes

On 308x, 4381, and 3090* Processor Complexes, the hardware requires its own I/O configuration data. Such data resides in input/output configuration data sets (IOCDS) in the processor controller. The processor controller can hold more than one IOCDS, allowing an installation to vary the I/O configuration according to the IOCDS. At system initial microprogram load (IML), the system operator chooses one IOCDS to define the I/O configuration.

Using the Input/Output Configuration Program, you can create a new IOCDS. VM/XA System Product requires its own definition of an I/O configuration. Such a definition comes from macro instructions

3090 is a trademark of the IBM Corporation.

contained in a file called HCPRIO ASSEMBLE. Be sure to make information in the HCPRIO ASSEMBLE file compatible with information in the IOCDS.

At system initialization, CP sends warning messages to the primary system operator about devices it finds offline. CP does this by checking the hardware I/O configuration defined by the IOCDS. If you have made sure that HCPRIO ASSEMBLE and the IOCDS are compatible, the primary system operator will receive warning messages only for devices that are intended to be online but were inadvertently left offline. System initialization procedures will therefore be less error prone.

If your computer center has more than one 308x, 4381, or 3090 Processor Complex on which VM/XA System Product may run, IBM advises you to create on each processor complex one IOCDS that is compatible with VM/XA System Product. Then you are always assured of having an I/O configuration compatible with VM/XA System Product.

For more information, see the *Input/Output Configuration Program User's Guide and Reference* for 308x concerns, the *IBM 4381 Input/Output Configuration Program User's Guide and Reference* for 4381 concerns, and the *3090 Input/Output Configuration Program User's Guide and Reference* for 3090 concerns.

VM/XA System Product Starter System

The starter system defines the following devices automatically:

- The primary system operator's console
- One DASD from which you will load (IPL) the VM/XA System Product starter system.

  Warning: Do not use your current system residence device. You will need it if for some reason you have to return to your current system.

The VM/XA System Product starter system allows you to define the other devices necessary for system generation (see Appendix C, "VM/XA System Product Starter System Information" on page 725). These devices are:

- The primary system operator's alternate console (optional, but recommended).
- One card reader and punch (not required).
- Two tape drives (the second tape drive is optional, but recommended).
- One printer.
- A DASD volume for use as a work pack. The work pack is required if you are using 3350 or 3375 DASD, or a single-density 3380 or 3390 DASD.

  Note: This volume must be the same device type and model as the system residence volume.
- A second work pack. The second work pack is required if you are using 3350 or 3375 DASD.

  Note: This volume must be the same device type and model as the system residence volume.
- A third work pack (with the system residence volume, there will be a total of four DASD volumes). The third work pack is required for the 3350 starter system. It is not available for the 3375, 3380, and 3390 starter systems.

  Note: This volume must be the same device type and model as the system residence volume.

Password Security

After you define the starter system configuration, you invoke the ITASK EXEC, which formats minidisks and loads files from the product tape. The first thing the EXEC does is to ask you for a password for MAINT and OPERATOR. You must supply a temporary password for MAINT and OPERATOR to log on and off the starter system. Table 1 on page 7 has a place for you to write in the temporary password—do so only if you take appropriate security measures, since the password allows access to the starter system.

A later step in the installation procedure tells you to update the user directory of your new system. Be sure to change all the passwords, including the passwords for MAINT and OPERATOR, to non-restricted passwords. (For the IBM-supplied list of restricted passwords, see Appendix F, "Restricted Logon Passwords" on page 859.)

Starter System Worksheet

Complete Table 1 before you begin the VM/XA System Product installation procedure. This table is designed to aid you throughout the installation procedure.

Table 1. Starter System Worksheet

Devices automatically defined by the starter system	**Write the device number (*rdevno*) of the devices**	**Circle the device type you will use for the installation procedure**
Primary system console		3277, 3278, 3279
Starter system IPL volume (XASRES)		3350, 3375, 3380, 3390
Devices you can define using the starter system	**Write the device number (*rdevno*) of the devices you are defining**	**Circle the device type you will use for the installation procedure**
Printer		1403, 3203, 3211, 3262, 3800-1 (see Note), 3800-3, 4245, 4248
Punch device (optional)		2540P, 3525
Card reader device (optional)		2501, 2540, 3505
First tape drive		3420, 3422, 3430, 3480
Second tape drive (recommended)		3420, 3422, 3430, 3480
Work pack (XASERV; required for all DASD except 3380-E4, 3380-K, and 3390)		3350, 3375, 3380, 3390
Second work pack (XAP001; required for 3350 and 3375 DASD)		3350, 3375, 3380, 3390
Third work pack (XAP002; required for 3350 DASD, not available for other DASD)		3350
Display device—alternate system console (optional)		3277, 3278, 3279
Password for MAINT and OPERATOR		Not applicable

Note: **3800** defaults to the 3800 Model 1.

Warning: The ITASK EXEC cannot handle virtual DASD that are defined with different physical characteristics from the real DASD. Consider this restriction when you define the virtual DASD used in the installation procedure. For example, do not define a real double-density 3380 as two virtual single-density 3380s.

First-Level and Second-Level Installation

The virtual machine concept of VM allows you to install VM/XA SP as the operating system of a real machine or as the operating system of a virtual machine.

Installation of VM/XA SP as the operating system of a real machine is called **first-level** installation. At first level, you use the system operator's console to control system functions.

The operating system usually run in a VM/XA SP virtual machine is CMS. However, other operating systems that can run in a virtual machine include VSE/SP, MVS/SP*, MVS/XA*, MVS/ESA*, VM/SP, VM/SP HPO, VM/XA SF, and VM/XA SP. Installation of VM/XA SP as the operating system of a virtual machine is called **second-level** installation. At second level, your "console" is a terminal logged on to the first-level VM system through a valid user ID.

When you do a first-level installation, you must have access to the real machine. You mount and ready the required DASD volumes and tapes, perform system IPLs (initial program loads), and display PSWs (program status words). This book assumes that you already know how to operate the computer and all of its associated hardware devices; no assistance is provided for these tasks.

When you do a second-level installation, you may have access only to your own virtual machine. Someone else, such as the computer operator, may have to mount and ready your DASD and tape volumes and attach them to your first-level user ID.

Note: You can install VM/XA System Product at second level only if your first-level VM/XA system has at least 16 megabytes of storage.

Perhaps the most important difference between first-level and second-level installation concerns system addresses. At first level, when you IPL your system residence volume, you IPL the real address of the DASD. When you IPL your system residence volume at second level, you IPL the virtual address of a minidisk (or DASD) attached to your first-level user ID. This first-level minidisk (which may be a full-pack minidisk) functions as a "real" DASD volume at second level.

At second level, it is also important to keep in mind that you can communicate with two levels of CP. First-level CP provides the virtual machine in which you are installing the VM/XA SP system that contains your second-level CP. At times when you are running at second level, you may want to issue commands to first-level CP. You can do this by first pressing PA1.

Because second-level users can issue commands to first-level CP, it is wise to limit the authority of the virtual machine that installs a second-level system. Specifically, this virtual machine should not have Class A authority so that it cannot shut down the first-level system.

MVS/SP, MVS/XA, and MVS/ESA are trademarks of the IBM Corporation.

Figure 4 shows the relationship between first-level and second-level operation.

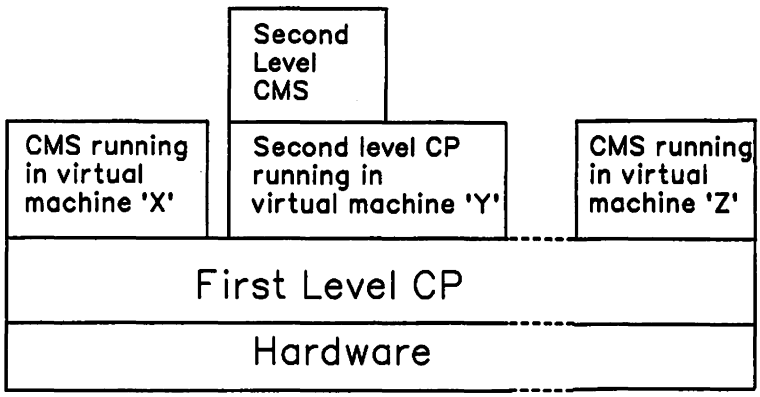


Figure 4. Relationship Between First-Level and Second-Level Operation

Planning for the Group Control System (GCS)

If you plan to install SNA (Systems Network Architecture) products or RSCS (Remote Spooling Communications Subsystem) Version 2, you must install the GCS component.

GCS is a named saved system that can be IPLed by a "group" of one or more virtual machines. GCS provides a variety of services to the group, such as supporting a native VM/SNA network that functions as part of your VM/XA SP system without help from a guest operating system.

For more information about GCS concepts and facilities, refer to the *VM/XA SP Group Control System Command and Macro Reference*, *VM/XA SP Features Summary*, and *VM/XA SP Planning and Administration.*

If you plan to install GCS, you must define the GCS saved system and the virtual machine group. A sample GCS saved system is already defined for you in the SAMPNSS EXEC, but you may want to tailor it to meet your own requirements. To define the virtual machine group, you invoke the GROUP EXEC to create a group configuration file.

You can install more than one GCS system on your VM/XA SP system. Planning for each GCS system involves the following operations:

- **Calculating your GCS storage requirements**

  You need to calculate how much private storage and common storage your GCS system requires. You use these figures during the installation procedure when you:

  - Tailor the GCS entry in the system directory
  - Change the GCS entry in the SAMPNSS EXEC
  - Create the GCS configuration file

  VM/XA SP Planning and Administration provides a procedure that you can use to calculate your storage requirements.

- **Planning the GCS entry in the system directory**

  The sample directory (USER DIRECT) supplied on the VM/XA SP product tape contains an entry for a GCS recovery machine (user ID GCS). Before you can use GCS, you **must** modify this entry to remove NOLOG from the USER statement and substitute a valid logon password. You may also have to change the machine size if you calculated a greater storage requirement.

In addition, if you have defined your GCS named saved system as restricted, you must add a NAMESAVE statement to the directory of each system allowed to use GCS.

VM/XA SP Planning and Administration explains the structure of system directory entries.

- **Planning the GCS entry in the SAMPNSS EXEC**

  When you have calculated your storage requirements for GCS, you can plan the GCS entry in the SAMPNSS EXEC. As supplied on the product tape, the SAMPNSS EXEC contains a sample DEFSYS entry for GCS. During the installation procedure, you can tailor this entry to define your GCS system.

  The SYSNAME that you specify in this entry (the supplied name is GCS) must be the same "system name" that you specify when you create the GCS configuration file.

  If you plan to install multiple GCS systems, you must create a new DEFSYS entry for each additional system and select a unique SYSNAME for each one.

  VM/XA SP Planning and Administration shows a sample DEFSYS entry.

- **Planning the GCS configuration file.**

  You create the GCS configuration file during the installation procedure by invoking GROUP EXEC, which presents a series of panels (screens) that ask you to supply certain information about your GCS system. You may find it helpful to record this information now in your Installation Reference Worksheet (space is provided to define up to three GCS systems).

  Note: If you do not have a full screen display device, you cannot use GROUP EXEC, because you cannot display the panels. In that case, you must build the configuration file manually during the installation procedure using the build macros described in the *VM/XA SP Group Control System Command and Macro Reference*. You may want to look at that book now to become familiar with the operation of those macros.

  System Name — This is the name of your GCS saved system. You must select a unique name for each GCS system that you want to install. GROUP EXEC creates a configuration file named *systemname* GROUP, which becomes the blueprint for your virtual machine group.

  Authorized Userids — These are the user IDs allowed to run in supervisor state and allowed to use GCS functions. For example, these user IDs have authorization to change common storage. Make sure that these user IDs have an appropriate privilege class specified in their directory entries.

  System Disk Address — This is the virtual address of the GCS system disk. MAINT 595 is the default, as defined in SPLOAD PROFILE and in the system directory.

  System Disk Ext. Address — This is the virtual address of the GCS system disk extension. MAINT 59E is the default, as defined in the system directory. This disk is available for use by applications.

  Common Dump Receiver — Normally, when a user ID's virtual storage is dumped to trace a problem, the dump file is spooled to the user ID's virtual reader. However, when you create the configuration file, you can select one authorized user ID to receive all GCS virtual storage dumps. If one of your GCS user IDs is the RSCS virtual machine, you **must** name a common dump receiver, since the RSCS virtual machine cannot process a dump file spooled to its own virtual reader.

  Recovery Machine — You **must** specify one authorized user ID to act as the recovery machine. The sample directory defines a recovery machine called GCS. (You may give it a different name.) The recovery machine must be the first machine to join your group, and is responsible for cleaning up system resources when other machines using those resources reset.

Trace Table Size	This table contains a history of GCS supervisor events. 16K is the default size. However, increasing the number of applications that you run also increases the activity of the GCS supervisor, which may require a larger trace table.
Max. Virtual Machines	This is the **maximum** number of virtual machines allowed in the group at one time. The number that you select depends upon how much space you have available in common storage.
System ID	This is the message that is displayed at the console of a virtual machine operator who IPLs this GCS segment. You can define up to 130 characters.
Saved Segments	These are the other saved segments (such as VTAM) that you want to access. They are linked automatically when you IPL your GCS system. Before you provide this information, you must find out what requirements these segments have. **Note:** Do **not** include CMSVSAM and CMSBAM segments here. Instead, follow the procedures described in Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" or Chapter 3, "Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)" to build these segments as part of your base installation.

Installing National Languages On Your VM/XA System Product Release 2.1 System

VM/XA System Product Release 2.1 is shipped with mixed-case American English as the system national language. This language is automatically set for CP and all virtual machines on the system. When users log on, they receive messages, see panels and HELP files, and enter CP and CMS commands in mixed-case American English.

At installation time, you can change the system default from mixed-case American English to uppercase American English.

You can also order other national languages to install on your system. National languages are distributed on language feature tapes. The files on a language feature tape contain translated information.

There are two ways to install a national language:

- As the new default system national language (replacing mixed-case American English or uppercase American English).

  Then, when users log on, they receive most messages, see some panels, and see some CP and CMS HELP files in the new language.

  In this type of language installation, the language files are used to create new system nuclei. The procedure is a supplement to the base installation procedure, and is described in Chapter 5, "Installing a New System National Language" on page 311.

  Notes:

  1. The extent of the translated information may vary between national languages, and some messages, panels, and HELP files may still appear in American English.
  2. The language of the installation and system generation tools is American English, and cannot be changed.

- As an option available to system users to supplement or override the system national language.

  You can install several national languages (including mixed-case American English and uppercase American English) in this manner, and you can set the languages that are available for individual users. This type of language installation must be performed **after** your base system is installed.

For more information, see *VM/XA SP Planning and Administration.*

Some Notes Before You Begin

Before you begin system generation, there are a number of things you must do. These things are:

- Check to see if you have the minimum hardware configuration required to generate VM/XA System Product. See Appendix C, "VM/XA System Product Starter System Information" on page 725.
- Use the Device Support Facilities to prepare DASD volumes.
- Refer to the *VM/XA System Product: Program Directory* for the latest information affecting VM/XA System Product.
- Be sure that you have the proper processor engineering change level for your processor. Consult your IBM representative to find out what it is.
- Make sure that you have no DASD volumes with the same labels as the installation work volumes (XASERV, XAP001, and XAP002). If you do, relabel them so that the starter system does not attach them to the system.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC must all match. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

Some Notes on the Installation Procedure

- Each installation process is divided into steps. In the event of a failure or termination for any reason, you can pick up the installation at the step immediately following the last successful step.
- The dialogs in the installation procedures are samples. Depending on your system configuration, the prompts and messages you see on your screen may not exactly match those in the book.

Chapter 2. Installing VM/XA System Product Release 2.1 with the Starter System (First Level)

Step 1. Load the Device Support Facilities

In this step, you will load the Device Support Facilities (DSF) and use it to initialize and inspect the DASD for defective tracks. For more information about the Device Support Facilities, see the *Device Support Facilities User's Guide and Reference*, SC35-0033.

This step takes approximately 20 to 40 minutes for each DASD volume you inspect, depending on the type of DASD.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Before you begin, make sure that you have read the information preceding these steps in Chapter 1, "Introduction" on page 3.
2. Ready the DASD volumes you plan to use. Follow the operation manual for your own hardware.

   If the packs you plan to use for your installation are not already ICKDSF-initialized, you must initialize them before proceeding.

   Warning: No volumes with labels XASERV, XAP001, or XAP002 should be attached to your system. Any such volumes will be brought on line when you IPL the starter system in Step 3. If necessary, you can use the Device Support Facilities to relabel your packs. Issue REFORMAT UNIT(*rdevno*) VERIFY(*oldlabel*) VOLID(*newlabel*) when DSF prompts you with `ENTER INPUT/COMMAND`.

   Warning: The addresses of XASRES, XASERV (if used), XAP001 (if used), and XAP002 (if used) must match the addresses in the HCPRIO ASSEMBLE file. If the addresses do not match, you will not be able to IPL your new system in Step 20. If your addresses are not defined in the sample HCPRIO ASSEMBLE file for your DASD type, you will have to tailor it in Step 17.
3. Mount the starter system tape on a tape drive.
4. IPL the tape drive. Follow the **hardware IPL** procedure specified for your processor. Refer to the proper hardware operation manuals for help. For instance, on the 3081 Processor Complex, refer to the *IBM 3081 Processor Complex Operator's Guide for the System Console.*

```
ICK005E DEFINE INPUT DEVICE, REPLY
'DDD,CUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:
console ■
ICK006E DEFINE OUTPUT DEVICE, REPLY
'DDDD,CUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:
console ■
ICKDSF - SA DEVICE SUPPORT FACILITIES 9.0 TIME:hh:mm:ss
  mm/dd/yy    PAGE 1
ENTER INPUT/COMMAND:
```

This message tells you that the Device Support Facilities are loaded and ready. **You MUST use the Device Support Facilities to format your XASRES volume unless it is already formatted.** If XASRES is already formatted, go on to Step 2.

```
init unit(rdevno) devtype(33xx) novfy val volid(volid) ■
```

rdevno is the address of the DASD volume you want to inspect. 33*xx* is the device type: 3350, 3375, 3380, or 3390. *volid* is the volume identifier, in this case, XASRES.

This command requests medial initialization of the volume. For maximal or minimal initialization, see the *Device Support Facilities User's Guide and Reference*.

```
ICK00700I DEVICE INFORMATION rdevno IS CURRENTLY AS FOLLOWS:
          PHYSICAL DEVICE = 33xx
          STORAGE CONTROLLER = xx
          STORAGE CONTROL DESCRIPTOR = xxxx
          DEVICE DESCRIPTOR = xx
ICK003D REPLY U TO ALTER VOLUME rdevno CONTENTS, ELSE T
ENTER INPUT/COMMAND
```

```
u ■
:
ENTER INPUT/COMMAND:
```

The system takes 20 to 40 minutes to inspect and reformat the DASD volume. After the DASD volume has been reformatted, you will get a series of ICK messages that describe the status of the device being formatted.

5. If you want to inspect or format any more DASD volumes, repeat the INIT UNIT command. Otherwise, go to Step 2.

Step 2. Restore the VM/XA System Product Starter System to Disk

In this step, you will use the DDRXA utility to restore the starter system to disk.

This step takes approximately 15 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. IPL the tape or cartridge. Follow the **hardware IPL** procedure specified for your processor. Refer to the proper hardware operation manuals for help. For instance, on the 3081 Processor Complex, refer to the *IBM 3081 Processor Complex Operator's Guide for the System Console.*

2. Establish an interactive dialog with DDRXA:

   Notes:

   a. When you run DDRXA on a real processor, and you have a nonconsole device at real device number 0009 or 001F, make sure the device is not operational. Otherwise, results are unpredictable. (DDRXA initially assumes that your console has device number 0009 or 001F).

   b. Also, to prevent inadvertent access to DDRXA, **turn off** all displays other than the one you will be using for DDRXA, so that no one will press the **ENTER** key on another display and interrupt DDRXA while it is waiting for input.

   As soon as you load it into storage, DDRXA begins executing and sends you the following initial prompt:

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:
```

   If the initial prompt is not shown on the display you are using for DDRXA, press the **ENTER** key. This action creates the necessary interrupt for DDRXA to recognize your display and to begin execution.

3. Answer the prompting messages from the DDRXA utility:

   Note: The device types for which the following steps are valid include 3350, 3375, 3380, and 3390.

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:
```

```
sysprint cons ■
ENTER:
```

   This first control statement tells DDRXA that you want program messages sent to your console.

```
input rdevno devtype (skip 1 ■
ENTER:
```

   The second control statement is the input control statement. Identify the device address (*rdevno*) and tape device type (3420, 3422, 3430, or 3480) where the starter system tape is mounted. The option **skip 1** tells DDRXA to skip over the tape mark placed on the tape just before the starter system.

```
output rdevno devtype scratch ■
ENTER:
```

This control statement specifies the mounted device number to which you are restoring the system (XASRES). Specify the device number (*rdevno*) and device type (*devtype*). Valid *devtype* values are:

3350
3375
3380
3390

The word **scratch** causes DDRXA not to check the label on the volume.

```
restore all ■
```

The RESTORE ALL statement tells DDRXA to restore the whole tape to the output device.

```
HCPDDR725D SOURCE DASD DEVICE WAS (IS) LARGER
              THAN OUTPUT DEVICE
           DO YOU WISH TO CONTINUE?
           RESPOND YES OR NO:
yes ■
```

Depending on the DASD model you are using, you may or may not see this prompt. If you do see it, respond **yes**. The starter system is designed to fit on all supported DASD models.

```
RESTORING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      cccc     cccc    cccc   cccc

⋮
END OF RESTORE
BYTES RESTORED nnnnnnnnnn
```

Informational messages: GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

```
ENTER:
■
END OF JOB
```

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

Step 3. Load the VM/XA System Product Starter System and Define Devices

In this step, you will:

- Load the VM/XA System Product starter system
- Define the devices necessary to perform the system generation.

This step takes approximately 10 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Load the VM/XA System Product starter system from the DASD device you just restored it to; that is, IPL the real address of XASRES. Follow the specified **hardware IPL** operation for your processor. Refer to the proper hardware operation manuals for help. For instance, on the 3081 Processor Complex, refer to the *IBM 3081 Processor Complex Operator's Guide for the System Console*. To prevent inadvertent access to the VM/XA starter system, power off all displays not to be used by the program.
2. If at first you receive no response, press the **ENTER** key on your display or other displays that are connected to the real processor. This action creates the necessary interrupt for the starter system to recognize your display and begin execution.
3. Use Table 1 on page 7 to answer the questions that the VM/XA System Product starter system asks you. The following are the starter system's informational messages and questions:

   Note: If you do not enter values, the system will use its default values. The system default is shown in the right-hand column. However, if the system defaults are not in your IOCDS, the system will not be able to use those devices.

```
DO YOU WISH TO REDEFINE YOUR SYSTEM (YES/NO):
yes ■
```

If you have not previously defined your system by executing this substep, answer **yes**. If you answer **no** your starter system may not IPL properly.

```
ENTER PRINTER NUMBER (NNNN):
rdevno ■
```

Enter the real device number of your printer. The system default is 000E.

```
ENTER DEVICE TYPE (1403, 3203, 3211, 3262, 3800, 3800-1, 3800-3, 4245, 4248):
devtype ■
```

Enter the device type of your printer. The system default is 1403.

Note: **3800** defaults to 3800 Model 1.

```
ENTER PUNCH NUMBER (NNNN):
rdevno ■
```

If you press **ENTER** without typing anything, you receive the system default, which is 000D. The starter system does not require a punch actually attached to it, but it does create a control block for a punch. If you want to use a real punch at a device number other than 000D, type in the device number and press **ENTER**.

```
ENTER DEVICE TYPE (2540P,3525):
devtype ■
```

If you press **ENTER**, you receive the system default, which is 2540P. The starter system does not require a punch actually attached to it, but it does create a control block for a punch. If you want to use a real punch other than the default, type in the device type and press **ENTER**.

```
ENTER READER NUMBER (NNNN):
rdevno ■
```

If you press **ENTER**, you receive the system default, which is 000C. The starter system does not require a reader actually attached to it, but it does create a control block for a reader. If you want to use a real reader at a device number other than 000C, type in the device number and press **ENTER**.

```
ENTER DEVICE TYPE (2540R, 2501, 3505):
devtype ■
```

If you press **ENTER**, you receive the system default, which is 2540R. The starter system does not require a reader actually attached to it, but it does create a control block for a reader. If you want to use a real reader other than 2540R, type in the device type and press **ENTER**.

```
ENTER NUMBER WHERE FIRST TAPE IS MOUNTED (NNNN):
rdevno ■
```

Enter the device number of the real tape device where the product tape will be mounted. The system default is 0580.

```
ENTER DEVICE TYPE (3420, 3422, 3430, 3480):
devtype ■
```

Enter the device type of your tape drive. The system defaults to a device type of 3420.

```
ENTER THE NUMBER OF A SECOND TAPE DRIVE (NNNN):
rdevno ■
```

You can type in a device number or just press **ENTER**, in which case you receive the default device number, 0581. The starter system does not require a second tape drive actually attached at 0581 or any other device number; the system does create a control block for the device. However, IBM recommends that you do have a second tape drive to send dumps to when the system is running.

```
ENTER DEVICE TYPE (3420, 3422, 3430, 3480):
devtype ■
```

Enter the device type of your second tape drive. The system defaults to a device type of 3420.

```
ENTER DEVICE NUMBER OF WORK PACK (NNNN):
rdevno ■
```

Enter the device number of your second pack (XASERV). The system default is 0230.

This pack is required for all DASD types **except** double- or triple-density 3380 DASD (3380-E4 and 3380-K) or 3390 DASD.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
devtype ■
```

Enter the device type you are using.

```
ENTER NUMBER WHERE EXTRA WORK PACK IS MOUNTED (NNNN):
rdevno ■
```

Enter the device number of your third pack (XAP001). **The 3350 and 3375 systems require this pack.** The 3380 and 3390 systems do not require an extra work pack.

Note: If you do not have XAP001, you will receive this message:

```
HCPLND108E MAINT 0124 NOT LINKED
```

when you log on as MAINT. If you are using 3380 or 3390 DASD, you can ignore this message.

If you press **ENTER**, you receive the system default, which is 0231.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
devtype ■
```

Enter the device type you are using.

3350 DASD Only

```
ENTER NUMBER WHERE EXTRA WORK PACK IS MOUNTED (NNNN):
rdevno ■
```

Enter the device number of your fourth pack (XAP002). **The 3350 system requires this pack.**

If you press **ENTER**, you receive the system default, which is 0232.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
3350 ■
```

End of 3350 DASD Only

```
ENTER NUMBER OF A DISPLAY DEVICE (NNNN):
rdevno ■
```

This step allows you to define an additional display device. This is **not** the console you are presently using. Do not take the default or use the address 021.

Check your installation's HCPRIO file to see if you have defined this display differently.

```
ENTER DEVICE TYPE (3277, 3278, 3279):
devtype ■
```

Enter the device type of your display. The system default is 3278.

```
***SYSTEM DEFINITION COMPLETED***
000E PRINTER
000D PUNCH
000C READER
0580 FIRST TAPE
0581 SECOND TAPE
0230 WORK PACK
0231 EXTRA WORK PACK
```

The numbers shown are those that would appear if you always take the default.

```
0232 OTHER WORK PACK
rdevno GRAPHIC DEVICE
ARE THE ABOVE ENTRIES CORRECT (YES, NO):
```

You will see this message only if you are using the 3350 starter system.

```
yes ■
```

Answer **yes** to this question. If you respond **no**, the procedure restarts at the first prompt of this step.

4. The Starter System continues:

```
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;

SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss,
  LOADED FROM XASRES
******************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
******************************************

HCPISU951I CP VOLID XASERV NOT MOUNTED
HCPISU951I CP VOLID XAP001 NOT MOUNTED
```

You may receive the *volume* not mounted messages.

```
START ((COLD|WARM|FORCE) (DRAIN)
  (DISABLE) (NODIRECT))|(SHUTDOWN)
cold drain ■
```

Since this is the first initialization of the system, do a COLD start with the DRAIN option.

Note: If the clock is not set, you may have to set the TOD clock using standard operating procedures. Consult *VM/XA SP Real System Operation* for those procedures.

```
NOW hh:mm:ss EST day mm/dd/yy
CHANGE TOD CLOCK (YES|NO)
yes ■
SET DATE MM/DD/YY
mm/dd/yy ■
SET TIME HH:MM:SS
hh:mm:ss ■
PRESS "TOD ENABLE SET" KEY AT DESIGNATED INSTANT
NOW hh:mm:ss EST day mm/dd/yy
CHANGE TOD CLOCK (YES|NO)
no ■
```

Note: If you are using a multiprocessor, you may receive a message here concerning the clocks of the different images of the processor. If you do, see *VM/XA SP Real System Operation* for information about resetting the clocks.

```
THE DIRECTORY ON VOLUME XASRES AT ADDRESS nnnn
   HAS BEEN BROUGHT ONLINE.
HCPLND108E MAINT 0124 NOT LINKED;
            VOLID XAP001 NOT MOUNTED
HCPLND108E MAINT 0125 NOT LINKED;
            VOLID XASERV NOT MOUNTED
```

CP logs on the primary system operator (user ID MAINT). These are informational messages for the real system operator. The FILES message refers to spool files; there are no spool files in the system, since this is a cold start of the system.

```
HCPLND108E MAINT 0126 NOT LINKED;
            VOLID XAP002 NOT MOUNTED
THERE IS NO LOGMSG DATA
FILES:      NO RDR,      NO PRT,      NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy
```

You will see this message only if you are using the 3350 starter system.

```
HCPIOP951I CP VOLID volid NOT
    MOUNTED
⋮
HCPIOP951I USER VOLID volid NOT
    MOUNTED
```

You may receive a number of informational messages that certain volume identifiers (CP *volid*s and user *volid*s) are not mounted. You may ignore these messages at this time.

```
STORAGE =  0016M
FILES: 0000001 RDR, 0000001 PRT, NO PUN
```

This message tells you the amount of real storage available. You must have 16 megabytes.

5. Initialize CMS and enable all displays:

```
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
```

The words VM READ appear in the bottom right corner of the display screen; when VM READ appears, press the **ENTER** key.

```
READY; T=n.nn/n.nn hh:mm:ss
```

```
enable all ■
READY; T=n.nn/n.nn hh:mm:ss
```

This command enables all displays available to the real system. The VM/XA System Product logo appears on all enabled displays.

6. Spool MAINT's console:

```
spool cons * start ■
READY; T=n.nn/n.nn hh:mm:ss
```

This command creates a spool file of all display output for the MAINT virtual machine. Spooling MAINT's console can be valuable if problems arise and you want to review the system generation activity. If you issue this command, you will have a record of this activity. When you close the file (SPOOL CONS CLOSE) or log off, the file is automatically sent to your reader.

7. Set the dump to a tape drive or printer address. It is recommended that you use the second tape drive you defined for sending dumps to. If you use a tape drive, mount a scratch tape with a ring on the drive.

set dump *rdevno* ■ {TAPE | PRINTER} *devno* DUMP UNIT CP IPL	Replace the *rdevno* with the real device number of your tape drive or printer to free up some of the spool space.

Warning: If you do not perform this step, you may run out of SPOOL space later on. In the event of a SHUTDOWN and restart at any time during this install procedure, you should perform this substep again.

Step 4. Attach the VM/XA System Product Release 2.1 Product Tape (Volume 1) and Load the First Three Files

In this step, you will:

- Mount Volume 1 of the VM/XA System Product product tape on a tape drive
- Attach the product tape drive to MAINT
- Load the first three files from the product tape.

This step takes approximately 5 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Mount Volume 1 of the VM/XA System Product product tape (in write-protect mode) on the tape drive that you specified as the first tape in "Step 3. Load the VM/XA System Product Starter System and Define Devices" on page 17. Follow the operation manual for the machine on which you mount the tape.

2. Attach the tape drive to MAINT at virtual device number 0181:

```
attach rdevno * 181 ■
TAPE rdevno ATTACHED TO MAINT 0181
READY; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the real device (*rdevno*) to MAINT's virtual machine at virtual device number 0181.

3. Check MAINT's disk accesses. Be sure MAINT's 191 minidisk is the A disk.

```
query search ■
MNT191   191 A    R/W
MNT190   190 S    R/O
READY; T=n.nn/n.nn hh:mm:ss
```

The QUERY SEARCH command tells you what disks you have accessed and the alphabetic order of those accesses. When searching for files, CMS uses the alphabetic order of accesses to create a search order. CMS searches the A minidisk, if accessed, then the B minidisk, if accessed, and so forth.

4. If MAINT's 191 minidisk is not A, access that minidisk as A:

```
access 191 a ■
191 REPLACES A (195)
READY; T=n.nn/n.nn hh:mm:ss
```

This message would appear if the QUERY SEARCH command had shown MNT195 195 A R/W.

5. For your own convenience, set **PF12** as a retrieve key. When you press the retrieve key, the last instruction you issued will be displayed. You need only press **ENTER** to enter it again.

```
set pf12 retrieve ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You may add this command to your PROFILE EXEC.

6. Use the VMFPLC2 LOAD command to load the first three files (the tape header, memos, and installation tools) from the product tape to your A-disk:

```
vmfplc2 load (eof 3 ■
 LOADING.....

 $TAPE$   HEADER   A1
 END-OF-FILE OR END-OF-TAPE
```

This command loads the first three tape files from the product tape. (A tape file is the set of files between two tape marks.) These are the only tape files you have to load—the ITASK EXEC automatically loads the rest of the tape. You will receive a message from VMFPLC2 telling you which files it loaded.

```
 SUPSP21  MEMO     A1
 56643089 SERVICE  A1
 END-OF-FILE OR END-OF-TAPE
```

The VMSUP memo is loaded to the MAINT 191 minidisk. Read this memo before going on.

After you perform Step 8, you may wish to copy SUPSP21 MEMO to the CP BASE disk (194 or 394) to free space on MAINT 191.

```
 56643089 $PPF     A1
 RPWLIST  DATA     A1
 $MSG4I$  EXCAMENG A1
 $$RDPW$$ EXEC     A1
 $MSG4I$  EXEC     A1
 SETUP    EXEC     A1
 SPLOAD   EXEC     A1
 UTILITY  EXEC     A1
 ITASK    EXEC     A2
 DIRECTXA MODULE   A2
 DVM      PROFILE  A1
 SPLOAD   PROFILE  A1
 END-OF-FILE OR END-OF-TAPE
READY; T=n.nn/n.nn hh:mm:ss
```

The tools in the third file are installation EXECs. You may erase them **after you finish the installation**, except for these files:

- DIRECTXA MODULE, which is erased during the installation procedure
- SETUP EXEC, which **must** remain on the MAINT 191 minidisk.

The DIRECTXA MODULE is loaded to MAINT 191 now so that you can bring your directory online in Step 5. It is erased after it is used, then reloaded to MAINT 190 in Step 8. If you need to reissue ITASK ALLOCATE, you must rewind the tape and issue VMFPLC2 LOAD DIRECTXA MODULE A (EOT to reload the DIRECTXA MODULE from Volume 1 of the product tape.

Step 5. Invoke the ITASK EXEC

In this step, you will use the ITASK EXEC to:

- Provide a password for user IDs MAINT and OPERATOR
- Load the sample directory and system definition files
- Format the XASERV, XAP001, and XAP002 volumes.

The time for this step depends upon the devices involved. It usually takes at least 30 minutes for each pack you format.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you haveon your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

Note: A description of the ITASK EXEC and its operands can be found in "ITASK EXEC" on page 611.

1. Issue ITASK ALLOCATE to supply a password for MAINT and OPERATOR and to format the XASERV, XAP001, and XAP002 volumes:

   Note: The example in this manual uses a 3380 DASD for the minidisks. If you use a 3350, 3375, 3380-E4, 3380-K, or 3390, your space allocations and starting address for CMS will differ.

   ```
   itask allocate ■
   ```

   ```
   REWIND COMPLETE
   DMSACC724I 191 REPLACES A (191)
   REWIND COMPLETE
   DMSWTK409I LOADING 33xx * TO 191 ATTACHED TO MAINT
   ```

   These are informational messages to tell you how the operation is progressing. *xx* is 50, 75, 80, 8E, 83, 90, or 91, according to your DASD type.

   Note: If ITASK exits with return code 1040, the 123 full-pack minidisk is not attached. 123 is the virtual address of XASRES. You may have to reattach XASRES to your system.

2. When prompted, enter a new password to be used as both your MAINT and OPERATOR passwords:

   ```
   ENTER ONE PASSWORD FOR MAINT AND OPERATOR:
   password ■
   ```

   You must supply a single password to be used for both the MAINT and OPERATOR user IDs. The ITASK EXEC will XEDIT the user directory and replace NOLOG with the logon password that you provide here.

   When selecting a password, refer to the RPWLIST DATA file shown in Appendix F, "Restricted Logon Passwords" on page 859. This file contains all of the restricted passwords for your VM/XA SP2.1 system. You cannot choose a password for MAINT and OPERATOR that exists in this file.

```
ENTER PASSWORD AGAIN TO VERIFY:
```

Reenter the password that you have designated for the MAINT and OPERATOR user IDs.

YOU NEED TO REMEMBER THIS PASSWORD FOR USE IN LATER STEPS. You can write it down on the worksheet—see Table 1 on page 7.

```
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION
   PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ON LINE
DMSWSL970I FORMATTING MAINT 193 MINIDISK
DMSWSL409I LOADING SYSGEN TOOLS TO THE 193 DISK ATTACHED TO MAINT
```

These first messages signify that the user directory has been created. Then ITASK loads the system generation tools needed next.

```
DMSWSL970I FORMATTING MAINT 295 MINIDISK
DMSWSL409I LOADING HCPSYSxx * TO THE 295 DISK
            ATTACHED TO MAINT
DMSWSL409I LOADING HCPRIOXA * TO THE 295 DISK
            ATTACHED TO MAINT
DMSWSL409I LOADING HCPBOX * TO THE 295 DISK
            ATTACHED TO MAINT
DMSWSL970I FORMATTING MAINT 395 MINIDISK
DMSWSL409I LOADING CMSLOC SAMPLES TO THE 395 DISK
            ATTACHED TO MAINT
DMSACC724I 191 REPLACES A (191)
```

These are messages from ITASK as it loads files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

```
DMSWTK965I YOU MAY WISH TO TAILOR THE FOLLOWING
           FILES BEFORE NUCLEI GENERATION:

           HCPRIO ASSEMBLE
           HCPBOX ASSEMBLE
           HCPSYS ASSEMBLE
           DMSNGP ASSEMBLE
           USER  DIRECT
```

You will be able to tailor HCPRIO ASSEMBLE, HCPBOX ASSEMBLE, and HCPSYS ASSEMBLE in "Step 17. Tailor the Sample HCPRIO, HCPSYS, HCPBOX Files and the Product Parameter File" on page 55. You will tailor DMSNGP ASSEMBLE in "Step 14. Tailor the DMSNGP Profile" on page 46. You will tailor USER DIRECT in "Step 21. Update the User Directory" on page 71.

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XASERV VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

rdevno or **skip** ■

Enter the real address, *rdevno*, of the XASERV volume. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you do not want to use XASERV, or if you have already formatted it, enter **skip**.

```
DASD rdevno ATTACHED TO MAINT 301
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 301

DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 301
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
CYLINDERS nnnn THROUGH nnnn FORMATTED
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XASERV'
```

This message signifies that CPFMTXA has completed formatting the XASERV volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD rdevno DETACHED MAINT 0301
DASD rdevno ATTACHED TO SYSTEM XASERV
```

The HCPFAM384I message signals the completion of this sequence.

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP001 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

```
rdevno or skip ■
DASD rdevno ATTACHED TO MAINT 302
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 302
```

Enter the real address, *rdevno*, to allocate the XAP001 volume. The 3350 and 3375 starter systems require XAP001. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you do not want to use XAP001, or if you have already formatted it, enter **skip**.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 302
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP001'
```

This message signifies that CPFMTXA has completed formatting the XAP001 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD rdevno DETACHED MAINT 0302
DASD rdevno ATTACHED TO SYSTEM XAP001
```

The HCPFAM384I message signals the completion of this sequence.

3350 DASD Only

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP002 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
rdevno or skip ■
DASD rdevno ATTACHED TO MAINT 303
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 303
```

Enter the real address, *rdevno*, to allocate the XAP002 volume. The 3350 starter system requires XAP002. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you have already formatted XAP002, enter **skip**.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 303
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP002'
```

This message signifies that CPFMTXA has completed formatting the XAP002 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD rdevno DETACHED MAINT 0303
DASD rdevno ATTACHED TO SYSTEM XAP002
```

The HCPFAM384I message signals the completion of this sequence.

End of 3350 DASD Only

```
READY; T=n.nn/n.nn hh:mm:ss
```

This message tells you that the ITASK EXEC has completed the sequence for each volume needed for installing VM/XA System Product Release 2.1.

Step 6. Format the Remaining Base CP Minidisks

In this step, you will use the ITASK EXEC to format the remaining minidisks that are defined in the CP directory, but have not yet been formatted or loaded in a previous step. ITASK also puts a PROFILE EXEC on the 191 minidisk for EREP and DISKACNT.

The time to perform this step may vary, but a good estimate is 45 minutes.

1. Use the ITASK EXEC with the BASEIDS parameter to format, load, and add (as appropriate) PROFILE EXECs to the remaining minidisks in the base CP directory. If you issue ITASK BASEIDS from a user ID other than MAINT, it will format that user ID's minidisks instead of MAINT's.

   Note: The block sizes of the 190 and 490 minidisks have increased from 1KB to 4KB for minidisk caching. The Starter System defines the 190 and 191 minidisks (formatting the 191 disk with a 1KB block size), while the ITASK BASEIDS step formats all other minidisks with a 2KB block size.

   If you choose to format a minidisk with a block size of 4KB, the number of cylinders for that minidisk may have to be increased, as is the case for the HELP disks.

```
itask baseids ■
DMSWTK1310I DO YOU WANT TO FORMAT MINIDISKS:
             AUTOLOG1 (191), CMSBATCH (195), DISKACNT (191),
             MAINT 36E (PVM), EREP (191), AND OPERATNS (191)?
DMSWTK8005R TYPE: 1 (YES) OR 0 (NO).
1 ■


DMSWTK968I THE FOLLOWING MINIDISKS
           DEFINED IN THE BASE CP DIRECTORY
           WILL BE FORMATTED:

           MAINT    292     MAINT    49C
           MAINT    194     MAINT    49D
           MAINT    19C     MAINT    501
           MAINT    19D     MAINT    591
           MAINT    19E     MAINT    592
           MAINT    201     MAINT    593
           MAINT    291     MAINT    594
           MAINT    293     MAINT    595
           MAINT    294     MAINT    596
           MAINT    391     MAINT    59E
           MAINT    392     MAINT    5E5
           MAINT    393     MAINT    691
           MAINT    394     MAINT    692
           MAINT    423     MAINT    791
           MAINT    490     MAINT    892
           MAINT    491     MAINT    895
           MAINT    492     MAINT    896
           MAINT    495     MAINT    89E

DMSWTK970I FORMATTING AUTOLOG1 191 MINIDISK
DMSWTK970I FORMATTING CMSBATCH 195 MINIDISK
DMSWTK970I FORMATTING DISKACNT 191 MINIDISK
DMSWTK969I A PROFILE EXEC HAS SUCCESSFULLY
```

```
      BEEN COPIED TO DISKACNT'S 191 MINIDISK
DMSWTK970I FORMATTING EREP 191 MINIDISK
DMSWTK969I A PROFILE EXEC HAS SUCCESSFULLY
      BEEN COPIED TO EREP'S 191 MINIDISK
DMSWTK970I FORMATTING MAINT 36E MINIDISK
DMSWTK970I FORMATTING OPERATNS 191 MINIDISK
DMSWTK970I FORMATTING MAINT 292 MINIDISK
DMSWTK970I FORMATTING MAINT 194 MINIDISK
DMSWTK970I FORMATTING MAINT 19C MINIDISK
DMSWTK970I FORMATTING MAINT 19D MINIDISK
DMSWTK970I FORMATTING MAINT 19E MINIDISK
DMSWTK970I FORMATTING MAINT 201 MINIDISK
DMSWTK970I FORMATTING MAINT 291 MINIDISK
DMSWTK970I FORMATTING MAINT 293 MINIDISK
DMSWTK970I FORMATTING MAINT 294 MINIDISK
DMSWTK970I FORMATTING MAINT 391 MINIDISK
DMSWTK970I FORMATTING MAINT 392 MINIDISK
DMSWTK970I FORMATTING MAINT 393 MINIDISK
DMSWTK970I FORMATTING MAINT 394 MINIDISK
DMSWTK970I FORMATTING MAINT 423 MINIDISK
DMSWTK970I FORMATTING MAINT 490 MINIDISK
DMSWTK970I FORMATTING MAINT 491 MINIDISK
DMSWTK970I FORMATTING MAINT 492 MINIDISK
DMSWTK970I FORMATTING MAINT 495 MINIDISK
DMSWTK970I FORMATTING MAINT 49C MINIDISK
DMSWTK970I FORMATTING MAINT 49D MINIDISK
DMSWTK970I FORMATTING MAINT 501 MINIDISK
DMSWTK970I FORMATTING MAINT 591 MINIDISK
DMSWTK970I FORMATTING MAINT 592 MINIDISK
DMSWTK970I FORMATTING MAINT 593 MINIDISK
DMSWTK970I FORMATTING MAINT 594 MINIDISK
DMSWTK970I FORMATTING MAINT 595 MINIDISK
DMSWTK970I FORMATTING MAINT 596 MINIDISK
DMSWTK970I FORMATTING MAINT 59E MINIDISK
DMSWTK970I FORMATTING MAINT 5E5 MINIDISK
DMSWTK970I FORMATTING MAINT 691 MINIDISK
DMSWTK970I FORMATTING MAINT 692 MINIDISK
DMSWTK970I FORMATTING MAINT 791 MINIDISK
DMSWTK970I FORMATTING MAINT 892 MINIDISK
DMSWTK970I FORMATTING MAINT 895 MINIDISK
DMSWTK970I FORMATTING MAINT 896 MINIDISK
DMSWTK970I FORMATTING MAINT 89E MINIDISK

DASD 333 DETACHED

READY; T=n.nn/n.nn hh:mm:ss
```

The ready message indicates that ITASK has completed its work successfully.

2. If any of the minidisks listed in message DMSWTK968I were not formatted, check your work packs to make sure that they are labeled XASERV, XAP001, and XAP002, and that they are attached to the system. Then reissue ITASK BASEIDS.
3. Continue with the next step.

Step 7. ITASK Loads Files from the Product Tape (Volume 1)

In this step, you will issue the ITASK EXEC to load files from Volume 1 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match.** If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Enter the ITASK command with the LOAD ALL1 operands to load the CP and dump viewing facility object code and service files. If a problem should occur during this step, or if for some reason you want to return to this step and load product tape files, see "ITASK EXEC" on page 611 to reload the particular files you need.

```
itask load all1 ■
DMSWSL409I LOADING SYSTEM SAMPLES TO
     THE 191 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP OBJECT TO
     THE 194 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP PUTAPPLY TO
     THE 292 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP PUTDELTA TO
     THE 294 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP CORAPPLY TO
     THE 292 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP CORDELTA TO
     THE 294 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW OBJECT TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW PUTAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW PUTDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW CORAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW CORDELTA TO
     THE 293 DISK ATTACHED TO MAINT
READY; T=n.nn/n.nn hh:mm:ss
```

These are messages from ITASK as it loads more files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

Step 8. ITASK Loads Files from the Product Tape (Volume 2)

In this step, you will issue the ITASK EXEC to load files from Volume 2 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match.** If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Mount Volume 2 of the product tape.
2. Issue ITASK LOAD CMS to load the CMS object code, source code, and service files and the IOCP files:

```
itask load cms ■
DMSWSL409I LOADING CMS BASE TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING IOCP FILES TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS PUTAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS PUTDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS CORAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS CORDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS SYSTEM TO
     THE 190 DISK ATTACHED TO MAINT
```

3. The ITASK EXEC invokes the SETUP EXEC to establish the minidisk access order, then the ASMGEND EXEC to build the system F assembler and create the associated auxiliary directory:

```
DMSACC724I 191 REPLACES A (191)
⋮
DMSACC724I 190 REPLACES A (191)
DMSACC725I 190 ALSO = S DISK
ASSEMBLE XF GEND PROC
ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED.
■
```

Press **ENTER** to place the modules on the S-disk, the system default.

Enter another disk-mode letter if you prefer the modules to be placed on another minidisk.

```
ASSEMBLE XF GEND COMPLETE

DMSACC724I 191 REPLACES A (190)
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

a. At this time, you may choose to copy the documentation file (SUPSP21 MEMO) to the CP BASE disk as suggested in Step 4, substep 6 on page 24.

b. You may choose to load HELP files, CP, CMS, and dump viewing facility source files, and GCS object files at this time. The following is an alternative order for loading these files that will minimize tape mounts:

   1) Load HELP files from Volume 2. (See Step 22 on page 72.)

   2) Load GCS object files from Volume 2. (See Step 26, substep 3 on page 78.)

   3) Load CP, CMS, and dump viewing facility source files from Volume 3. (See Step 23 on page 73.)

Step 9. Establish the Service Tools Build Disk

In this step, you copy the latest level of the service tools to the service tools build disk.

This step takes approximately 10 to 15 minutes.

To be sure that you are using the most recent service EXECs, you must have the latest level of service on a service tools build disk. This disk, 5E5, is defined in the sample directories. It is the only disk in the TASK string in the product parameter file.

1. IPL 190:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order and create a work EXEC to help you copy the service tools:

```
setup ■
Ready; T=n.nn/n.nn hh:mm:ss
DMSACC724I vdevno replaces fm (vdevno
⋮
vmfsetup 56643089 cms ■
DMSACC724I vdevno replaces fm (vdevno
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
xedit lvlses exec fm ■
input ■
```

fm is the filemode of the task disk (5E5).

```
&TRACE OFF
&FT = &LEFT OF &2 2
&FT = &CONCAT OF &FT *
EXEC FILELIST &1 &FT *
&EXIT 0
```

Enter these lines exactly as they appear.

```
■
■
file ■
```

Press **ENTER** twice to return to the command line.

3. Look for the latest version of the CPYSVSES EXEC or CPYSVSES EXC*nnnn*. CPYSVSES contains a list of all the service EXECs.

```
filelist cpysvses * * ■
```

Examine each version of CPYSVSES. CPYSVSES has an update history at the top. The latest level is one with the most updates. Use the update history, **not the date and time stamp,** to determine which is the most recent version.

4. Copy the latest level of CPYSVSES to the task disk (5E5) with a filetype of EXEC:

```
copyfile / = exec fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

5. Invoke CPYSVSES EXEC:

```
cpysvses exec lvlses ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. CPYSVSES EXEC invokes LVLSES, which invokes FILELIST to display each entry for each service EXEC listed in CPYSVSES. Process each FILELIST screen as follows:

   a. For each of the files on the FILELIST screen, determine which is the latest version. **Do not depend on the date and time stamps;** examine the actual files. Each file ends with an update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn*, MOD*nnnnn*, XED*nnnnn*, and PRF*nnnnn* are exact images of the corresponding EXEC, MODULE, XEDIT, and PROFILE executables.

   b. Copy the latest files to the task disk (5E5), using the OLDDATE REPLACE options. Assign the copy a filetype of EXEC, MODULE, XEDIT, or PROFILE:

```
copyfile / = ft fm (olddate replace ■
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

   c. Press **PF3** to exit this FILELIST screen and go to the next screen.

   d. Continue until there are no more FILELIST screens.

7. Look for the latest version of the product parameter file:

```
filelist 56643089 * * ■
```

Examine each version of the product parameter file. A sample product parameter file is shown in "A Sample Product Parameter File" on page 384. The product parameter file has an update history at the top. The latest level is the one with the most updates. Use the update history, **not the date and time stamp,** to determine which is the most recent version.

8. Copy the latest level of the product parameter file to the task disk (5E5):

```
copyfile / = $ppf fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

9. Now that the service tools are all on the service tools build disk, you do not need to issue SETUP again. Instead, simply access 5E5 as B. You may wish to add this access to MAINT's PROFILE EXEC.

```
access 5E5 b ■
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase any duplicate files from MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

`filelist * * b` ■	This command lists all the files on your B-disk.
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.
`erase` ■	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the B-disk (5E5) or any other disk on which it may appear.

Step 10. Set the System Default National Language

In this step, you will set the system default national language.

This step takes approximately 20 minutes.

The system default national language is mixed-case American English. This means that most system messages and HELP files will appear in mixed case (capital and small letters). Some will appear in uppercase (capital letters only).

Decide whether you want to keep the system default as mixed-case American English or change it to uppercase American English (in which case all system messages and HELP files will be in uppercase). Follow the appropriate instructions below.

If you want to install a system default national language other than American English, you will be able to do so after you finish the installation process. See Chapter 5, "Installing a New System National Language" on page 311 for instructions.

If you want to install an alternate system national language (including whichever version of American English is not your default), see *VM/XA SP Planning and Administration.*

Mixed-Case American English

If you want to keep the system default national language as mixed-case American English, you should convert the installation messages to mixed case. (If you do not do so, all the installation messages will continue to appear in uppercase, but most other system messages will appear in mixed case.)

1. Save the uppercase installation message file:

```
rename $msg4i$ exec a = excuceng = ■
READY; T=n.nn/n.nn hh:mm:ss
```

2. Rename the mixed-case installation message file:

```
rename $msg4i$ excameng a = exec = ■
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSAMENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $MSG4I$ EXEC, the service will be in uppercase. You do not have to convert it to mixed case.

Uppercase American English

If you want to change the system default to uppercase American English, you must:

1. Edit the product parameter file and change all :NLS. tags from AMENG to UCENG:

```
xedit 56643089 $ppf ■
ch/:NLS.      AMENG/:NLS.      UCENG/* * ■
file ■
```

There are six blanks after :NLS. in the XEDIT change command.

2. Save DMSNGP TEXT, then rename the DMSNGP TXTUCENG file DMSNGP TEXT. This gives you a TEXT file assembled from a DMSNGP profile with LANGID=UCENG and HELP=19C.

   Note: You will be able to make other changes in the DMSNGP profile in Step 14, after you install Assembler H. You will have to change the LANGID= and HELP= parameters in Step 14. The DMSNGP TXTUCENG file is provided here only to allow you to specify uppercase American English as your system default national language.

```
access 395 d ■
DMSACC724I 395 REPLACES D(395)
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp text d = txtameng = ■
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp txtuceng d = text = ■
READY; T=n.nn/n.nn hh:mm:ss
```

3. Convert $VMFMSG$ EXEC (the service message file) to uppercase:

```
xedit $vmfmsg$ exec ■
uppercas * ■
file ■
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSUCENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $VMFMSG$ EXEC, the service will be in mixed case. You will have to convert it to uppercase. If you receive service for the DMSNGP profile or the product parameter file, you will have to update them again.

Step 11. Build CMS

In this step, you will build CMS.

This step takes approximately 5 minutes.

1. IPL 190 and issue ITASK BUILD CMS to invoke the CMS BUILD process:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss

spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss

itask build cms ■
nnnnnnnn files changed

DMSACC724I 191 replaces A(191)
DMSACC724I 5E5 replaces B(5E5)
```

ITASK invokes the SETUP EXEC to create the correct CMS access order.

```
DMSWSU1900W The existing 56643089 $SETUP A1
            file has been refreshed.  You
            might want to check your access
            order when done.
```

You will see this message only if you have a copy of 56643089 $SETUP on your A-disk. If you do see it, you will get return code 4.

```
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

A temporary load list is created and the system loader is invoked.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
  RECS nnnK COPY 001 NOHOLD NOKEEP
```

The CMS nucleus has been sent to your reader.

```
DMSWSU1906E The access of 19E failed with a
            return code of 28.  Processing
            stops for product 56643089.
            The original access order cannot
            be restored.
```

The access failed because 19E is empty. You may receive this message. It is not a problem. You will reaccess 19E when you re-IPL 190. If you see this message, you will get return code 4.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF which is now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

2. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

3. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand asks for a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 7 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

4. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

5. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    KEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

6. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

7. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

```
HCPLDR8013I Possible overlay:  .SLC 000000
```

If you see either of these messages, you can ignore them. They are informational messages.

```
DMSINS283E The NLSlangid saved segment could not be found;
            return code 44 from SEGMENT
```

```
DMSINQ609R Nucleus (CYL or BLK) address =
cyl ■
```

Enter the correct starting cylinder address for the DASD type of your 190 minidisk:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

```
DMSINQ612R Enter version identification:
■
```

Press **ENTER** for the default version identification, VM/XA CMS 5.6 *mm/dd/yy hh:mm*. You will be able to modify the default in Step 14.

```
DMSINQ612R Enter installation heading:
■
```

Press **ENTER** for the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM. You will be able to modify the default in Step 14.

```
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

Informational message: The optional installation segment (CMSINST) is not loaded and saved until "Step 29. Create the CMSINST and HELP Saved Segments" on page 103 is completed.

8. You may want to load the printer image library at this time. If you do, see "Step 25. Install the Printer Image Library" on page 75.

Step 12. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to MAINT's virtual printer. Ready the printer to send the CMS load map to MAINT's virtual reader:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt * ■
```

```
RDR FILE fileno SENT FROM MAINT
PRT AS fileno RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The CMS load map has been sent to MAINT's virtual reader. If you do not get this message, it has already been sent there by a previous IPL 190.

2. If the CMS load map is sent to MAINT's virtual printer instead of the reader, transfer it to the virtual reader:

```
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Identify the reader file:

```
query rdr * all ■
```

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8100 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

4. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft ■
File fn ft D received from MAINT at * sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

5. Print a copy of the CMS load map. If you do not have a printer attached, skip this substep.

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Examine the load map for errors. Verify that the new updates have been included in the CMS nucleus:

```
xedit fn ft ■
⋮
qquit ■
```

7. Continue with the next step.

Step 13. Install Assembler H Version 2 Program Product

Note: IBM packages Assembler H Version 2 as a separate program product. This program product has its own installation procedure.

In this step, you will:

- Consult *Assembler H Version 2 Program Product: Installation*, SC26-4030
- Install Assembler H Version 2 Program Product, VM* feature
- Install service for Assembler H Version 2.

The time to perform this step may vary, but a good estimate is 20 minutes.

1. Before you install Assembler H, consult your IBM Support Center for the latest information on APARs.
2. Read the *Installation* manual for Assembler H Version 2 Program Product.
3. Refer to "Installing Program Products" in the *VM/XA SP2.1 Program Directory* for more information on Assembler H.
4. Follow the directions in the *Installation* manual to install Assembler H Version 2. IBM advises you to install Assembler H Version 2 on the 19E minidisk.

   Note: Use the OBJECT option to assemble Assembler H CSECTS.
5. During the installation of Assembler H Version 2, answer YES to the question "BUILD AN AUXILIARY DIRECTORY?" and respond Y to the question asking what mode letter you want to use to access the disk containing the H-Assembler modules.
6. Erase IEV61 TEXT. If you do not erase this file, you will not be able to apply service to Assembler H.
7. Install the latest service level of Assembler H.
8. After you have finished installing service for Assembler H, IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

VM is a trademark of the IBM Corporation.

9. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase these duplicate files from MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

`filelist * * fm` ■	This command lists all the files on the task disk (5E5).
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.
`erase` ■	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the task disk (5E5).

10. If any unneeded service memos exist on the MAINT 191 disk, erase them to make room for further processing.

Step 14. Tailor the DMSNGP Profile

In this step, you will use the VM System Product Editor (XEDIT) supplied with the starter system to tailor the CMS nucleus generation profile (DMSNGP).

This step, excluding time for printing system definition files, takes approximately 15 minutes.

1. Before proceeding with this step, read the discussion of the DMSNGP profile in *VM/XA SP Planning and Administration*, under the discussion of the DEFNUC macro. DMSNGP ASSEMBLE is a profile that contains CMS configuration defaults and responses to system prompts; these will become part of the CMS nucleus.
2. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Access the CMS LOCAL1 minidisk, where DMSNGP resides, as D:

```
access 395 d
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

In this example, the DMSNGP file resides on the 395 disk.

4. Save a copy of DMSNGP ASSEMBLE, so that if you make a mistake in changing it, you can go back to the original file:

```
copyfile dmsngp assemble d = oldassem = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. XEDIT the CMS nucleus generation profile:

```
xedit dmsngp assemble ■
```

Within the file, you can use the normal XEDIT cursor-control keys, scroll keys, insert key, and delete key.

```
NGP      TITLE 'DMSNGP     (CMS)      VM/XA SYSTEM PRODUCT 5664-308'
         EJECT
DMSNGP   CSECT
         DEFNUC  SYSDISK=190,         * S disk address                *
                 YDISK=19E,           * Y-disk address                *
                 HELP=19D,            * Help disk address             *
                 LANGID=AMENG,        * Default is American English   *
                 DBCS=NO,             * Default is not a DBCS lang    *
                 LANGLEV=S,           * DCSS ID for multiple DCSS     *
                 SAVESYS=NO,          * Using CMS in DCSS yes or no   *
                 SYSNAME=CMS,         * Name of above DCSS to save    *
                 USEINST=YES,         * Using EXEC/XEDIT in DCSS      *
                 INSTSEG=CMSINST,     * Name of above DCSS to save    *
                 REWRITE=YES,         * Write nucleus yes or no       *
                 IPLADDR=190,         * Address of where to write     *
                 CYLADDR=?,           * CYL/BLK OF WHERE TO WRITE     *
                 IPLCYL0=YES,         * Write IPL text on cyl 0       *
                 VERSION=?,           * VM/XA CMS 5.6 MM/DD/YY HH MM SS*
                 INSTID='VM/XA CONVERSATIONAL MONITOR SYSTEM'
         END
```

6. Look at the DEFNUC defaults, and make any necessary changes. Refer to *VM/XA SP Planning and Administration* for information on the DEFNUC macro.

   Warning: The comma at the end of each line of the DEFNUC macro is required in all lines but the last line. If you omit this comma in any line other than the last line, all the lines after the first line without a comma are ignored.

   Warning: On each line, make sure that you leave at least one blank between the comma and the comment.

   - At a minimum, you should change **CYLADDR=?**, to indicate the starting cylinder on the MAINT 190 minidisk at which to start writing the CMS nucleus. The appropriate starting cylinder depends on the device type of the DASD where the MAINT 190 minidisk is defined:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

     If you do not replace the question mark for the CYLADDR keyword, you receive the following prompt when you generate the CMS nucleus:

     ```
     DMSINI609R Nucleus (CYL or BLK) address =
     ```

     You must respond with the appropriate cylinder address.

   - If you specify **YES** after the SAVESYS option, you must create a skeleton named saved segment (NSS) before building the CMS nucleus. This segment is saved when you build CMS and IPL your reader. (You will get message HCPNSS440I.)

     A skeleton segment for the CMS nucleus is created by the SAMPNSS EXEC. You can also create your own skeleton segment with the DEFSYS command. For information on the DEFSYS command, see *VM/XA SP CP Command Reference* and *VM/XA SP Virtual Machine Operation.*

- Make sure the LANGID and HELP parameters are right for your system default national language:

Language	LANGID	HELP
Mixed-case American English	AMENG	19D
Uppercase American English	UCENG	19C

- Make any other changes you wish. For example, you may want to change the version identifier or the installation header. You must put **single quotes** around your version identifier and installation header. The character string between the single quotes can be any combination of alphanumeric characters.

```
file = = fm ■
```

Make your changes and file the update file on the CMS LOCAL1 (395) disk.

7. Assemble the new DMSNGP file:

```
itask assemble dmsngp ■
⋮
DMSUPD181E No update files were found
DMSWHM1907I Assembling DMSNGP
```

The minidisk access order is reestablished.

Note: If an error occurs during the assembly, examine the flagged statements and correct the statements in error. Then continue from the beginning of this step.

```
DMSWHM1909I DMSNGP TEXT A created
PRT FILE fileno SENT FROM MAINT   PUN AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

The DMSNGP file has been assembled and placed on the MAINT 191 minidisk.

8. Copy the text deck from MAINT 191 to MAINT 395, then erase it from MAINT 191:

```
copy dmsngp text a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase dmsngp text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the filemode of the CMS LOCAL1 minidisk.

If you did not replace the question mark for the CYLADDR keyword, when you generate the CMS nucleus, you will receive the following message:

Message	Appropriate Response
DMSINI609R Nucleus (CYL or BLK) address =	*nnnnn* *nnnnn* is the location on the 190 disk where the new CMS nucleus is written. Refer to the table on page 47 to determine the correct response for your DASD type.

Step 15. ITASK EXEC Rebuilds the CMS Nucleus

Note: Do this step only if you have changed the DMSNGP file in Step 14.

In this step, the ITASK EXEC:

- Creates a CMS nucleus
- Places the CMS nucleus on the CMS residence disk.

This step takes approximately 5 minutes.

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Issue ITASK BUILD CMS to create the CMS nucleus. By issuing the ITASK EXEC, you will incorporate updated CMS text files into a new CMS nucleus.

```
itask build cms ■
nnnnnnnn files changed
⋮                                          ITASK invokes the SETUP EXEC to create the
                                           correct CMS access order.
DMSWSU1900W The existing 56643089 $SETUP A1   You will see this message and return code 4 only if
            file has been refreshed.  You     you did not erase 56643089 $SETUP in Step 11.
            might want to check your access
            order when done.
⋮                                          A temporary load list is created, and the system
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)       loader is invoked.
RDR FILE fileno SENT FROM MAINT PUN AS fileno   The CMS nucleus has been sent to your reader.
  RECS nnnK COPY 001 NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss                If you get return code 4, the reason may be that a
                                           PTF now being applied has another PTF, outside
                                           CMS, as a prerequisite. Check the $VMFBLD
                                           $ERRLOG file for message DMSBNC1854W.
```

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■                      The ALL operand requests a display of all
                                           information about the reader files.
```

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file you will IPL in substep 8 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
HCPLDR8015I POSSIBLE OVERLAY:  .SLC 000000
```

You may receive this message. It can be ignored.

CMS Nucleus Generation Prompts and Responses

Note: Each of the following prompts appears **only** if the corresponding statement in DMSNGP is missing or empty (and the DEFNUC macro contains no default value), or if the DMSNGP statement contains a question mark (?):

```
DMSINQ606R System disk address =
■
```

190 is the default.

```
DMSINQ615R Y-disk address =
■
```

19E is the default.

```
DMSINQ640R HELP disk address =
■
```

19D is the default.

```
DMSINQ764R Language id =
■ or langid ■
```

This response identifies the *langid* of your system national language. The default *langid* is AMENG (mixed-case American English).

```
DMSINQ293R Is this a DBCS language?  Enter 1 (YES) or 0 (NO).
```

0 ■

The default is 0 (NO); mixed-case American English is not a DBCS (Double-Byte Character Set) language.

```
DMSINQ295R Language level id =
```

■

The system national language does not use a level ID.

```
DMSINQ296R Should the installation segment be used?  Enter 1 (YES) or 0 (NO).
```

The installation segment is an optional shared saved segment, into which you can place frequently used EXECs and System Product Editor (XEDIT) macros. You install the segment **after** you install your base system, but you must indicate now whether or not you are going to use it.

■ (or **1** ■) or **0** ■

The default is 1 (YES). Enter **0** if you do not want to use the segment.

```
DMSINQ310R Installation segment name =
```

This prompt appears only if you accepted the default (or entered **1**) at the previous prompt.

■ or *segname* ■

Enter a 1- to 8-alphanumeric-character name for the installation segment, or press **ENTER** to accept the default name, CMSINST.

```
DMSINI729R Do you want to save the system?  Enter 1 (YES) or 0 (NO).
```

0 ■

Answer **0** (NO). You will save CMS in Step 27.

The default is 1 (YES). **Warning:** The default response to this prompt does not agree with the sample DMSNGP file, where SAVESYS=NO.

```
DMSINI730R Saved system name =
```

You will not see this prompt if you answered **0** to the preceding prompt.

■ or *sysname* ■

The default system names are CMS and CMSXA.

```
DMSINI607R Rewrite the nucleus?  Enter 1 (YES) or 0 (NO).
```

1 ■

Enter **1** to write the CMS nucleus on the disk that you specify in your response to the next prompt.

```
DMSINI608R IPL device address =
```

■

The default is the address of the system disk (190).

```
DMSINI609R Nucleus (CYL or BLK) address =
```

nnn ■

nnn is the location on the 190 system disk where the new CMS nucleus is written. Enter the correct cylinder address for your DASD type:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

```
DMSINI610R Also IPL CYL/BLK 0?  Enter 1 (YES) or 0 (NO).
1 ■
```

Enter **1** to write the initial IPL text on cylinder/block 0 of the disk where the CMS nucleus is written.

The initial IPL text is a bootstrap program that reads the CMS nucleus from the cylinder/block where the nucleus is written (as defined in your response to prompt DMSINI609R). The initial IPL text is always written on the same cylinder/block as the nucleus. If the initial IPL text is not also written on cylinder/block 0, you must specify the cylinder/block address of the nucleus when you issue IPL commands for this system. For more information, refer to the description of the IPL command in *VM/XA SP CP Command Reference.*

```
DMSINI611R Enter version identification:
■ or version ■
```

The version identification is displayed each time that you IPL the CMS system you are now generating.

You can enter up to 32 descriptive characters to identify this version and level of CMS, or you can press **ENTER** to accept the default version identification, VM/XA CMS 5.6 *mm/dd/yy hh:mm.*

```
DMSINQ612R Enter installation heading:
■ or heading ■
```

The installation heading appears at the beginning of each output file created using this CMS nucleus.

You can enter up to 64 descriptive characters to serve as an installation heading, or you can press **ENTER** to accept the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM.

End of CMS Nucleus Generation Prompts and Responses

```
DMSINS327I The installation saved segment could not be loaded
```

Informational message: The CMSINST installation segment is not loaded and saved until you complete Step 27 on page 91.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

This is the default version identification. If you defined your own version identification, it appears here and each time that you IPL 190 or IPL CMS.

```
⋮
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 16. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to MAINT's virtual printer. Ready the printer to send the CMS load map to MAINT's virtual reader:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt * ■
```

```
RDR FILE fileno SENT FROM MAINT
PRT AS fileno RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The CMS load map has been sent to MAINT's virtual reader. If you do not get this message, it has already been sent there by a previous IPL 190.

2. Identify the reader file:

```
query rdr * all ■
```

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,300 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

3. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft d ■
File fn ft D received from MAINT at * sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1−9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

4. To print a copy of the CMS load map, issue:

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Examine the load map for errors:

```
xedit fn ft ■
⋮
qquit ■
```

6. Continue with the next step.

Step 17. Tailor the Sample HCPRIO, HCPSYS, HCPBOX Files and the Product Parameter File

In this step, you will:

- Edit the real I/O configuration and system definition sample files
- Tailor the HCPBOX sample file if you wish to change either the default logo that appears on your screen when you log on the system or the default logo that appears on separator pages
- Tailor the product parameter file only if you have a reason for changing the supplied CP nucleus generation information
- Assemble the real I/O configuration file and the system definition files.

The time to perform this step varies depending upon the time you spend editing files.

Overview

Note: If you are applying updates with AUX files, follow the service procedures in Part 2, "Servicing the System."

As shipped on the product tape, the product parameter file and the sample files contain sample information and default parameters. Based on the requirements that were established in pre-installation planning, **you must examine and modify these files** to define your unique system configuration.

The product parameter file, 56643089 $PPF, contains information that the system generation tool (VMFBLD EXEC) uses when building system nuclei.

The sample files, which you loaded from the product tape to MAINT 295 in "Step 5. Invoke the ITASK EXEC" on page 25, contain the following information:

- The real I/O configuration file (HCPRIO ASSEMBLE) defines the configuration of your system input/output devices.
- The CP system control file (HCPSYS ASSEMBLE) describes the system residence device (XASRES) and various system parameters.
- The HCPBOX ASSEMBLE file defines both the logo that appears on your screen when you log on your system and the logo that appears on separator pages.

Note: You may find it useful to print copies of the system files for future reference. These include HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and your user directory (default USER DIRECT).

To tailor any of these files, use the VM/SP System Product Editor (XEDIT). XEDIT offers full-screen editing on display terminals.

Note: When you XEDIT ASSEMBLE files, issue SET TRUNC 72 so that you do not lose the continuation character in column 72.

Procedure

1. Access 295 as D:

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. See Appendix C, "VM/XA System Product Starter System Information" on page 725 for the sample files that correspond to your VM/XA System Product starter system, and "A Sample Product Parameter File" on page 384 for a sample product parameter file. Unless you alter them, the ITASK EXEC will use these files as they appear in Appendix C, "VM/XA System Product Starter System Information" when it creates the new CP nucleus. If you wish to alter the real I/O configuration file, the system definition files, or the product parameter file, follow the instructions in this step.

HCPBOX ASSEMBLE

If you want to change the design or contents of either the logo that appears when you log on to the system, or the logo that appears on separator pages, tailor the sample HCPBOX ASSEMBLE file. **Do not edit HCPBOX ASSEMBLE.** IBM services this file and bases the service on the version supplied. Use the procedure for local modifications.

1. Establish the appropriate minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you receive a return code of 4 and a DMSWSU1900W message. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
vmfsetup 56643089 cp (all ■
erase 56643089 $setup a ■
```

2. Verify that HCPXA CNTRL lists an auxiliary control file for local changes:

```
xedit hcpxa cntrl ■
TEXT MACS HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT  AUXPAT TX$  * LOCAL PATCHES
LCL  AUXLCL LCL  * LOCAL MODIFICATIONS
TEXT AUXXA       *
qquit ■
```

HCPXA CNTRL should have a line like this one. The second field is the filetype of the auxiliary control file that points to local modifications.

3. Edit the auxiliary control file:

```
xedit hcpbox auxlcl ■
input ■
```

You will add your local modifications at the top of the file.

```
R00001LC LCL LC00001  *UPDATE HCPBOX LOGO ■
```

In this example. **R00001LC** is the filetype of the update file, and **LC00001** is the local tracking number. The rest of the line, beginning with the asterisk (*), is a comment explaining the purpose of this modification. The text deck created when you assemble HCPBOX with your local modifications will be called HCPBOX LCL00001.

```
■
file = = fm ■
```

Press **ENTER** to return to the command line.

fm is the CP LOCAL1 disk (295).

4. Create the update file and apply the updates:

```
xedit hcpbox assemble (ctl hcpxa ■
```

Instead of editing HCPBOX ASSEMBLE directly, you will apply changes to a copy of it.

```
DMSXUP178I Applying HCPBOX AnnnnnHP
```

The updates supplied by IBM for HCPBOX ASSEMBLE are being applied.

```
DMSXUP180W Missing PTF file HCPBOX R00001LC F5
```

Your update file cannot be found because you have not created it yet.

```
 HCPBOX R00001LC A1
      |...+....1....+....2....+....3....+
===== * * * Top of File * * *
=====    .
=====    .
=====    .

====>
```

The system copies the source file and allows you to edit the copy.

```
set serial off ■
set trunc 72 ■
⋮
file = = fm ■
```

You want the source file's sequence numbers and continuation characters to remain unchanged.

Make your changes and file the update file on the CP LOCAL1 (295) disk.

HCPRIO ASSEMBLE

You can use a procedure similar to the one shown for updating HCPBOX ASSEMBLE to update HCPRIO ASSEMBLE, or you can use the procedure shown below.

1. Edit the real I/O configuration source file. This file is on MAINT's 295 minidisk. The file is called HCPRIO ASSEMBLE.

   Use the XEDIT command to edit the file. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   You **must** make sure that your system volumes and consoles are defined properly in HCPRIO. If they are not, you will not be able to IPL your new system.

   For the real I/O configuration file, you may want to:

   - Delete those RDEVICE macro instructions for device numbers that the system will not be likely to use
   - Add RDEVICE macro instructions for device numbers that do not appear in the sample file.

Tailor HCPRIO ASSEMBLE to match the IOCDS, which is the data set that defines the I/O configuration for the hardware. See "System Installation on 308x, 4381, and 3090 Processor Complexes" on page 5.

Note: When tailoring HCPRIO, minidisk caching is not allowed for shared DASD. To control sharing on a device level, use the RDEVICE SHARED=YES option in HCPRIO or use the SET SHARED ON FOR *rdev* command. These will effectively turn off minidisk caching on a device basis. We recommend that you set the RDEVICE macro in HCPRIO rather than using the SET SHARED ON FOR *rdev* command.

See *VM/XA SP Planning and Administration* for details about HCPRIO macro instructions and for a high-level explanation of HCPRIO ASSEMBLE.

2. File the edited version of the real I/O configuration file on MAINT's 295 minidisk. Use the XEDIT subcommand FILE.

HCPSYS ASSEMBLE

You can use a procedure similar to the one shown for updating HCPBOX ASSEMBLE to update HCPSYS ASSEMBLE, or you can use the procedure shown below.

1. Edit the system definition source file. The file, called HCPSYS ASSEMBLE, is on MAINT's 295 minidisk.

   Use the XEDIT command to edit the file. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   For the system definition file, you may want to:

   - Change the device number for the system residence device on the SYSRES macro instruction.

     Note: By convention, IBM assigns device number 0123 for the system residence device. This device number appears in four places:

     - On the DIRECTORY control statement in the user directory.
     - On an MDISK control statement in MAINT's entry in the user directory (this gives MAINT read/write access to the system residence device).
     - On the SYSRES macro instruction.
     - In the product parameter file. (The device number for the system residence device might not be included in the product parameter file [56643089 $PPF].)

     If you change the device number in one place, be sure to make an identical change in the other three places.

   - Modify the SYSSTORE macro instruction. Set the RMSIZE operand to the storage of your processor. If your system will have a virtual=real area (including storage for virtual=fixed guests), set the VRSIZE operand. If any virtual machines will need to issue DIAGNOSE X'98' in System/370* mode, or in 370-XA mode with 24-bit addressing, set the RIO370 operand.
   - Modify the SYSTIME macro instruction for your time zone.

System/370 is a trademark of the IBM Corporation.

- Tailor the volume identifiers on the SYSCPVOL and SYSUVOL macro instructions to match the volume identifiers your system will use. Because this action eliminates extraneous warning messages, the action helps the real system operator identify volumes that are not mounted at IPL time.
- Supply your own system identifier on the SYSID macro instruction.

See *VM/XA SP Release Guide* for Release 2.1 for details on HCPSYS macro instructions.

2. File the edited version of the system definition file on MAINT's 295 minidisk. Use the XEDIT subcommand FILE.

56643089 $PPF

1. The default 56643089 $PPF should be appropriate for most installations, but if your experience suggests changes, you can override 56643089 $PPF.

   DO NOT alter this file unless you have a special need.

   The best way to change the product parameter file is to create an override file. For a discussion of override files, see "The Product Parameter Override File" on page 402.

   Use the XEDIT command to create an override file on the TASK (5E5) minidisk. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC must all match. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

2. File the override file on the TASK (5E5) minidisk. Use the XEDIT subcommand FILE.

Assemble the Files

1. Assemble the HCPRIO, HCPSYS, and HCPBOX ASSEMBLE files:

```
itask assemble filename ■
⋮
DMSWHM1907I Assembling filename

DMSUPD178I Updating HCPBOX ASSEMBLE B1
DMSUPD178I Applying HCPBOX AnnnnnHP F5
DMSUPD178I Applying HCPBOX ft B5
DMSWHM1909I filename {TEXT|ft} A created
RDR FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

If an error occurs while assembling any of the system definition files, see "Correcting Assembly Errors" on page 60.

You will see these messages only when you assemble HCPBOX ASSEMBLE, and only if you updated it.

2. Copy these text files from your A-disk to the CP LOCAL1 (295) minidisk:

   HCPSYS *ft*
   HCPRIO *ft*
   HCPBOX *ft*

```
copy filename {text|ft} a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase filename {text|ft} a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the CP LOCAL1 minidisk (295).

Correcting Assembly Errors

To correct assembly errors in system definition ASSEMBLE files, do the following:

1. Examine the flagged statements. See *VM/XA SP Planning and Administration* for directions on coding.
2. Correct the statements causing an error by editing the system definition file with the System Product Editor (XEDIT).
3. Reassemble the system definition file in which the change was made using the procedure in "Assemble the Files" on page 59.
4. Repeat the sequence until all files assemble correctly.

Step 18. Generate the New CP Nucleus

In this step, you will use the ITASK EXEC to build your new CP nucleus.

This step takes approximately 20 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Purge all unnecessary files from your reader to increase available spool space.
2. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the ITASK EXEC to build the CP nucleus.

   Note: Before you issue ITASK BUILD, ensure that you are running in 370 mode by doing the following:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

   Now continue with the substep:

```
itask build cp noassem ■
```

Use the **noassem** only if you have already assembled HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and HCPBOX ASSEMBLE. If you have not assembled these files, you **must** omit the **noassem** operand, so that ITASK will assemble them now.

If you assemble the files here, copy the text decks from MAINT 191 to the CP LOCAL1 minidisk (295), then erase them on MAINT 191.

```
nnnnnnnn FILES CHANGED
DMSACC724I 191 replaces A(191)
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

RDR FILE fileno TO MAINT COPY 001 NOHOLD
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The CP nucleus is being created. It will be sent to your reader.

If you get return code 4, the reason may be that a PTF which is now being applied has another PTF, outside CP, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

4. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

5. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 10 on page 63 below. It has approximately 75,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

6. Order your reader so that the CP nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CP nucleus.

7. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED   NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

9. If necessary, redefine your virtual storage. First, find out how much virtual storage MAINT has:

```
query virtual storage ■
STORAGE = nnnnM
Ready; T=n.nn/n.nn hh:mm:ss
```

Now determine how much storage you need:

- You need at least 16 megabytes to IPL your virtual reader in substep 10 on page 63.
- If you are generating a CP nucleus with a preferred virtual machine, MAINT's virtual storage must be at least 4 megabytes greater than the sum of the VRSIZE, VRFREE, and RIO370 operands in the SYSSTORE macro instruction in HCPSYS ASSEMBLE. For more information on the SYSSTORE macro instruction, refer to *VM/XA SP Planning and Administration*.

  If MAINT's virtual storage must be set to a value greater than 32MB (the maximum size for MAINT as defined by the sample USER DIRECT file), then you must tailor the directory to change the maximum virtual machine size for MAINT. Once the directory is changed, bring it online by

issuing DIRECTXA *directory-name*. If this is not done, you will receive wait state 8028 when the reader is IPLed.

If you need to increase your storage to more than 16 megabytes, first issue:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

Storage addresses greater than 16M require 370-XA architecture. If your machine is already in 370-XA mode, you will not get a response.

If you need to increase your storage (even if you need only 16M), issue:

```
define storage nnnnm ■
STORAGE = nnnnM
STORAGE CLEARED - SYSTEM RESET
```

10. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

```
hh:mm:ss * MSG FROM MAINT    :
  HCPGEN9010W NUCLEUS LOADED ON XASRES
  HCPGIR450W  CP ENTERED; DISABLED WAIT
  PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus was successfully loaded on the system residence device.

If you receive disabled wait state code 8028, see "Correcting IPL Errors."

If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

11. Issue the following commands to ensure that these settings are in effect:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
```

If your machine is already in 370 mode, or if your storage is already 16M, you will not get a response.

You have now loaded the CP nucleus.

Correcting IPL Errors

If you receive disabled wait state code 8028 when you try to IPL your reader, you must correct the problem as follows:

1. Close the reader. Issue CLOSE RDR.
2. Determine the reader file number of the IPL deck. Issue QUERY RDR ALL.
3. Take the IPL deck out of USER HOLD status. Issue CHANGE RDR *nnnn* NOHOLD, where *nnnn* is the reader file number of the IPL deck. If this step is not done, you will receive wait state 232 when the reader is re-IPLed.
4. Tailor the directory (default USER DIRECT) as described in substep 9 on page 62.

5. Reissue DIRECTXA *directory-name*.
6. IPL the reader. Issue IPL C CLEAR.

Correcting Load Errors

If the new CP nucleus fails to load, do the following:

1. IPL CMS:

```
ipl 190 clear ■
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWSP327I The installation saved segment could not be loaded

VM/XA CMS 5.6 mm/dd/yy hh:mm                    If you have changed the version heading, your
■                                               own heading will appear.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Access 295, then inspect the CP load map and virtual PSW. An error in loading may show up in either the load map or in the PSW.

   For a list of loader wait-state codes, refer to *VM/XA SP System Messages and Codes Reference*.

3. Correct the error.
4. Go back to the point where the error may have occurred. If you cannot determine where the error may have occurred, see "Step 13. Install Assembler H Version 2 Program Product" on page 44.

Step 19. Save and Print the CP Load Map

In this step, you will:

- IPL CMS
- Save the load map of the new CP system
- Print a copy of the new CP load map.

This step takes approximately 10 minutes.

1. When you built CP, the CP load map was spooled to MAINT's virtual printer. The load map must be transferred to MAINT's virtual reader. If you have IPLed 190 since building CP, the load map may have already been spooled from the virtual printer to the virtual reader.

   Query the printer to see if the load map is still there. If it is, note the file number. If it is not, skip substep 3 and go to substep 2.

   ```
   query prt all ■
   OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
   MAINT  fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

2. IPL CMS:

   ```
   ipl 190 clear ■
   DMSWSP327I The installation saved segment could not be loaded
   VM/XA CMS 5.6 mm/dd/yy hh:mm
   ■
   SYNONYM SYN
   CP TERM MODE VM
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

CP Load Map in Virtual Printer Only

3. Transfer the printer file to your reader:

   ```
   transfer prt fileno * rdr ■
   RDR FILE fileno TRANSFERRED FROM * FILE fileno
   0001 FILE TRANSFERRED
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

End of CP Load Map in Virtual Printer Only

4. Access the CP LOCAL1 disk:

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the disk on which you will receive the load map.

5. Identify the reader file:

```
query rdr * all ■

ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST    USERFORM OPERFORM KEEP MSG
MAINT    fileno A PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG STANDARD STANDARD OFF  OFF
```

The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

6. Bring in the file from the reader to the CP LOCAL1 disk:

```
receive fileno fn ft d ■
fn ft D1 created
DMSRDC738I Record length is 132 bytes
File fn ft D received from MAINT at *
  sent as (none) (none) D
```

This command saves the load map on the CP LOCAL1 (295) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

7. To print a copy of the load map for the CP nucleus, issue:

```
spool 00c class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The load map is printed on your system printer.

8. Examine the load map for errors:

```
xedit fn ft ■
⋮
```

9. Examine the load map for unresolved symbols. There should be no unresolved symbols.

```
locate /unresol ■                          You are still in XEDIT mode.
DMSXDC546E Target not found
qquit ■
```

Step 20. Load the New CP Nucleus

In this step, you will:

- Shut down the VM/XA System Product starter system
- Perform a hardware initial program load of the new CP nucleus.

This step takes approximately 10 minutes.

1. Shut down the starter system you are now using. A class A user (usually the primary system operator) must issue the SHUTDOWN command.

```
shutdown ■
SYSTEM SHUTDOWN COMPLETE
```

2. IPL the real address of your system residence device. Follow the hardware operations guide for your processor.

   If you are unable to IPL your system residence device, go back to "Step 2. Restore the VM/XA System Product Starter System to Disk" on page 15 and continue from there.

```
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;
SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss,
LOADED FROM XASRES
******************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
******************************************
⋮
HCPISU951I CP VOLID volid NOT MOUNTED
```

   Note: A 9025 disabled wait state indicates that your system volume (XASRES) is not defined in the HCPRIO ASSEMBLE file. A 1010 disabled wait state indicates that your console is not defined in the HCPRIO ASSEMBLE file.

3. During the initialization phase, respond **cold drain** to the START message:

```
START ((COLD|WARM|FORCE) (DRAIN) (DISABLE) (NODIRECT))|(SHUTDOWN)
cold drain ■
```

```
NOW hh:mm:ss EDT day mm/dd/yy
CHANGE TOD CLOCK (YES|NO):
no ■
```

Note: If the clock is not set, set the TOD clock using standard operating procedures. Consult *VM/XA SP Real System Operation* for those procedures.

```
The directory on volume XASRES at address nnnn
   has been brought online.
There is no logmsg data
FILES: NO RDR, NO PRT, NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy
```

```
HCPCPJ951I CP volid volid not mounted
⋮
```

volid is a DASD volume listed on the SYSCPVOL macro instruction of HCPSYS. If the volume is not one that you are using, ignore the message.

You are now logged on to the OPERATOR user ID.

```
STORAGE = 0016M
FILES 0000001 RDR, 0000001 PRT, NO PUN
```

```
XAUTOLOG AUTOLOG1
HCPAUT053E AUTOLOG1 not in CP directory
XAUTOLOG EREP
HCPAUT053E EREP not in CP directory
XAUTOLOG DISKACNT
HCPAUT053E DISKACNT not in CP directory
```

After you have modified the user directory, these error messages will no longer appear.

4. Define the storage that you want for the OPERATOR user ID:

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
```

This is an example. You may define more than 16M.

5. Enable the devices in your revised HCPRIO file:

```
enable all ■
hh:mm:ss Command complete
terminal mode vm ■
```

6. IPL 190:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

```
DMSACP1123E A(191) device error
Ready; T=n.nn/n.nn hh:mm:ss
```

If you see this error message, you have not formatted the operator's 191 minidisk.

7. Format the operator's 191 minidisk **only if you have not already done so:**

```
format 191 a ■
DMSFOR603R FORMAT will erase all files on disk A(191).
           Do you wish to continue?
           Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
opr191 ■
DMSFOR733I Formatting disk A
DMSFOR732I n cylinders formatted on A(191)
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Disconnect from OPERATOR and log on as MAINT.

   Note: If you do not have XAP001, you will receive this message:

   ```
   HCPLND108E MAINT 0124 NOT LINKED
   ```

   when you log on as MAINT. If you are using 3380 or 3390 DASD, you can ignore this message.

Step 21. Update the User Directory

In this step, you will:

- Edit the sample user directory
- Use the DIRECTXA command to update the user directory.

The time to perform this step varies depending on the time you take to edit files.

1. See Appendix C, "VM/XA System Product Starter System Information" on page 725 for sample user directories.

   Note: All logon passwords are NOLOG and there are no MDISK passwords, except for some of MAINT's minidisks. You may want to change these.

2. Edit the sample user directory (USER DIRECT on MAINT's 295 minidisk):

```
access 295 d ■
xedit user direct ■
⋮
```

   Be sure to change the logon passwords in USER DIRECT from NOLOG to your own non-restricted installation passwords. (For a list of restricted passwords, see Appendix F, "Restricted Logon Passwords" on page 859.) Also, be sure to change the logon password you provided for the MAINT and OPERATOR virtual machines in "Step 5. Invoke the ITASK EXEC" on page 25. This password appeared in the console file.

   Also, take this opportunity to add or change MDISK passwords for the MDISKs defined in USER DIRECT.

   Note: The read, write, and multiple-write passwords are positional. Refer to the skeletal MDISK statement provided in USER DIRECT. If you will be starting accounting and error recording later, this is a good time to add or change MDISK passwords for the DISKACNT and EREP 191 minidisks.

   For more information about the user directory, see *VM/XA SP Planning and Administration.*

3. When your changes are complete, enter **file** on the command line.

4. Use the DIRECTXA command to update and place the user directory online:

```
directxa user direct ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ON LINE
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 22. ITASK Loads HELP Files from the Product Tape (Volume 2)

In this step, the ITASK EXEC loads HELP files from Volume 2 of the product tape.

The time this step takes varies from system to system, but a good estimate is 40 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match.** If If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 2 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD AMENGHLP, LOAD UCENGHLP, or LOAD HELP operands to load the HELP files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK command again.

   ```
   itask load {amenghlp|ucenghlp|help} ■
   ```

   The HELP files on the tape are in mixed-case American English:

   - Use the **amenghlp** operand to load the HELP files to the 19D disk.
   - Use the **ucenghlp** operand to load the HELP files directly to the 19C disk, then convert them to uppercase. Conversion takes about 20 minutes.
   - Use the **help** operand to load the HELP files to the 19D disk, then copy them to the 19C disk and convert the copied files to uppercase.

   ```
   DMSWSL409I Loading AMENGHLP FILES to the
              19D disk attached to MAINT
   ```

   You will see this message if you used the **amenghlp** or **help** operand.

   ```
   DMSWSL409I Loading UCENGHLP FILES to the
              19C disk attached to MAINT
   ```

   You will see this message if you used the **ucenghlp** operand.

   ```
   CONVERTING HELP FILES TO UPPERCASE
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   You will see this message if you used the **ucenghlp** or **help** operand.

Step 23. ITASK Loads Source Files from the Product Tape (Volume 3)

In this step, the ITASK EXEC loads CP, CMS, and dump viewing facility source files from Volume 3 of the product tape.

The time this step takes varies from system to system, but a good estimate is 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match.** If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 3 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD ALL3 operands to load all the source files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK LOAD ALL3 command again.

```
itask load all3 ■
DMSWSL409I Loading CP SOURCE to the
           394 disk attached to MAINT
DMSWSL409I Loading CMS SOURCE to the
           393 disk attached to MAINT
DMSWSL409I Loading DUMPVIEW SOURCE to the
           393 disk attached to MAINT
```

These are messages from ITASK as it loads files from the product tape.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

ITASK has finished restoring source files from Volume 3 of the product tape.

Note: Source files are **not** unpacked during the load.

Step 24. Install Environmental Record Editing and Printing

In this step, you will:

- Read the documentation for the Environmental Record Editing and Printing (EREP) Program
- Follow the directions to install EREP.

The time to perform this step may vary, but a good estimate is 15 minutes.

Note: IBM packages the Environmental Record Editing and Printing (EREP) Program, VM Feature, 5654-260, as a separate program product. This program product has its own installation procedure.

1. Read the documentation for EREP:
   - The following memos in tape files 1 and 2 of the VMSUP EREP tape:
     - 5654260B MEMO
     - F5654260 MEMO2
     - I5654260 MEMO
     - 5654260B SERVICE
   - The Program Directory for EREP, which describes the installation procedure for EREP.

     Note: This document does not come with the VMSUP tape. You must order it separately.
   - The discussion of EREP in *VM/XA SP Planning and Administration.*
2. Follow the directions in the EREP *Program Directory* to install EREP from the VMSUP EREP tape.

   Note: You will receive messages HCPCRC8059I and HCPCRC8060I until EREP is installed.
3. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 25. Install the Printer Image Library

In this step, you will install the image library for your printer and save it on tape.

This step takes approximately 10 minutes.

Warning: Do not attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.

1. Load (IPL) 190. This is the CMS system disk.

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you receive a return code of 4 and a DMSWSU1900W message. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
access 5E5 b ■
vmfsetup 56643089 cp ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the IMAGELIB command. The command requires the filename of the image library (*fn* CNTRL). IBM supplies default image library control files called IMAG*xxxx* CNTRL, where *xxxx* is the printer number: 3800, 1403, 3203, 3211, 3262, 4245, or 4248.

   For more information about image libraries, see *VM/XA SP Planning and Administration.*

```
imagelib libname ■                         If you use the IBM-supplied image library,
HCPNMT247I NAMED IMAGE IMAGxxxx CREATED    libname is IMAGxxxx, where xxxx is the printer
Ready; T=n.nn/n.nn hh:mm:ss                number.
```

4. Check the status of the image library:

```
query img all ■

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*IMG     nnnn IMG  A  nnnn mm/dd hh:mm:ss fn       IMG      MAINT
```

Check the current state of the file (CL), the filename, and filetype.

5. Issue the appropriate LOADBUF command and start the printer. See *VM/XA SP CP Command Reference* for more information.
6. Mount a scratch tape with ring on an available 3420, 3422, or 3430 tape drive, or insert a cartridge into an available 3480 unit, with the thumbwheel turned until the white dot is not visible. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.
   - The tape is suitable for a 3420, 3422, 3430, or 3480 tape unit.

   Note: This procedure assumes the tape mode is 1600 bits per inch.
7. Issue the SPTAPE command to dump the image library to tape:

```
sptape dump rdevno img all run ■
```

Substitute the real device number of the tape drive for the value *rdevno*. The operand RUN specifies that SPTAPE rewinds and unloads the tape after the operation.

```
DUMPING rdevno IMG *IMG    nnnn A
⋮
SPTAPE DUMP FUNCTION ON
   DRIVE rdevno COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The message from SPTAPE tells you the file is being dumped to tape.

8. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE LOAD 181 IMG ALL RUN command to restore the image libraries. For more information about the SPTAPE command, see *VM/XA SP Real System Operation.*
9. If necessary, reattach your tape drive.

Step 26. Load, Build, and Save GCS

You must install the GCS (group control system) component if you plan to install SNA products or RSCS Version 2. **Before you begin, read "Planning for the Group Control System (GCS)" on page 9.**

If you do not want to install GCS, go to "Step 28. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments" on page 95.

To install GCS, you will do the following steps:

- Load GCS from Volume 2 of the product tape.
- Invoke the GROUP EXEC and complete a series of panels to create a GCS configuration file.

  Note: If you do not have a full-screen display device, you cannot use the GROUP EXEC because you cannot display the panels. In that case, you must build the configuration file manually, using the build macros described in the *VM/XA SP Group Control System Command and Macro Reference.*
- After you complete the panels, you invoke the ITASK EXEC with the BUILD GCS parameters.

  ITASK:
  - Modifies a copy of the GCS loadlist (GCSLOAD EXEC) and changes the entry that contains the filename of the GCS configuration file (the default is GCS) to match the filename of the configuration file that you just created (the default is also GCS).
  - Files the modified loadlist on the MAINT 295 minidisk. The modified loadlist is used during the generation of the GCS named saved system.
  - Renames the filetype of the GCS configuration file from GROUP to ASSEMBLE.
  - Invokes VMFHASM EXEC to assemble the configuration file.
  - Invokes VMFBLD EXEC with the BUILD GCS parameters to build and save the GCS nucleus.
- After you build and save the nucleus, you can save and print the GCS load map, which contains the storage addresses of the nucleus control sections and entry points.
- If you want to install more than one GCS nucleus, you must re-access the GCS minidisk structure and re-invoke the GROUP EXEC to create another GCS configuration file. Each configuration file must have a unique name and have an entry in the SAMPNSS EXEC.

This step takes approximately 30 minutes for each GCS system.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC must all match. If you change an address in any one of these places, you must change it in all the others. We recommend that you do not change any addresses.

1. Attach a tape drive to MAINT.
2. Mount Volume 2 of the product tape.

3. Issue ITASK LOAD GCS to load the GCS code:

```
itask load gcs ■
DMSWSL409I LOADING GCS INTERFACE TO
     THE 190 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS OBJECT TO
     THE 595 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTDELTA TO
     THE 596 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORDELTA TO
     THE 596 DISK ATTACHED TO MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

4. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Establish the correct minidisk order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Set up the GROUP EXEC messages in storage:

   Notes:

   a. If you are REBUILDING a GCS nucleus and your current configuration file does not require changes, you can skip this part of the step and go directly to the nucleus build operation on page 86.

   b. If you do not have a full-screen display device, you **cannot** use the GROUP EXEC to build the configuration file. **Do not** set up the GCS messages in storage. Instead, refer to the *VM/XA SP Group Control System Command and Macro Reference* and use the build macros described there to build the configuration file manually. Then continue this step with the nucleus build operation on page 86.

```
copyfile csimes[y] [text|txtnnnnn] fm csiume[y]text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

y is the country code for your system default national language: blank for mixed-case American English and **b** for uppercase American English. You can use QUERY LANGLIST to determine your system default national language.

The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a field that reads UM*nnnnn*, where *nnnnn* is the PTF number.

Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*.

You will have one with a filetype of TEXT.

This command creates a temporary GCS message file (containing the messages for the GROUP EXEC) that has the filename required by the SET LANGUAGE command. A permanent message file will be created when you generate the GCS nucleus. If there has been any service to the message file, the permanent file will reflect it.

```
set language langid (add csi user ■
Ready; T=n.nn/n.nn hh:mm:ss
```

langid identifies your system default national language. At this point, *langid* will be either **ameng** (mixed-case American English) or **uceng** (uppercase American English).

This command sets the language of the temporary GCS message file and places the file in user storage.

7. Invoke the GROUP EXEC to display the configuration panels.

   Use your installation reference worksheet to help you complete the panels. For more information, see "Planning for the Group Control System (GCS)" on page 9.

```
group systemname ■
```

This command assigns *systemname* as the filename of the GCS configuration file that you are creating and invokes the Primary Option Menu. The *systemname* parameter is optional at this time. If specified, it must match the system name in the DEFSYS entry for this GCS system in the SAMPNSS EXEC (the sample system name is GCS). If you specify a system name here, the Primary Option Menu appears with the system name filled in. If you do not specify a system name here, you must specify one on the Primary Option Menu.

Table 2. Function Keys Used with the GROUP EXEC Panels	
Key	**Function**
PF1 HELP	Shows information about the panel you are looking at.
PF2 CLEAR	Clears the input areas where you enter information.
PF3 END	Leaves the present panel and returns you to a previous one. • If you press **PF3** on the primary option menu, you return to CMS. • If you press **PF3** on any other screen, you return to the Primary Option Menu. • If you press **PF3** after you have entered information, but have not saved it, you receive the message: `577E File has been changed; type QQUIT to quit anyway`
PF5 REFRESH	Fills in the panel's input areas with the values you last saved there.
PF6 SAVE	Saves information you've entered on the panel (for the configuration file *systemname* GROUP).
PF7 PREVIOUS	Returns to the previous panel (if any).
PF8 NEXT	Moves ahead to the next panel (if any).
PF9 VERIFY	This PF key is not supported by VM/XA SP.
PF12 CURSOR	Moves the cursor to the panel's command line.
PF4, **PF10**, **PF11**	Not used.
ENTER	Processes any valid CP or CMS command typed on the command line. Two specific commands you can enter are: 1. QQUIT, entered from the primary option menu, returns you to CMS. If entered on any other panel, it returns you to the Primary Option Menu. 2. CANCEL, entered on any panel, returns you to CMS.

```
 GRP1                  GCS GROUP - PRIMARY OPTION MENU                  Primary
------------------------------------------------------------------------------

Fill in the blanks with the required information and then press the ENTER key.

       Type/change the name of the saved system that is being defined.

                        SYSTEM NAME : ________

          Type one number from the list below to display/update the:

                     1.  Authorized VM Userids.
                     2.  Saved System Information.
                     3.  Saved Segment Links.

             Type your choice here: _


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  ...      6  ...
  PF: 7  ...      8  ...      9  VERIFY   10  ...    11  ...     12  CURSOR

====>
```

8. To complete the Primary Option Menu:

   a. **Fill in or change the SYSTEM NAME.**

      If you invoked GROUP with the *systemname* parameter, this panel appears with the SYSTEM NAME already filled in.

   b. **Select the next panel (1, 2, or 3) and press ENTER.**

      The first time you leave the Primary Option Menu, select panel 1; the next time, panel 2; the third time, panel 3.

```
GRP11          AUTHORIZED VM USERIDS FOR <     >       PAGE 1 OF 1
------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the authorized VM userids.
       To CHANGE, type a new userid over the userid to be changed.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.

        ________   ________   ________   ________________
        ________   ________   ________   ________________
        ________   ________   ________   ________________
        ________   ________   ________   ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY   10  ...    11  ...      12  CURSOR

====>
```

9. To complete the Authorized Userids panel:

   a. **Type in the user IDs.**

      Press **ENTER** or **PF6** after typing the user IDs to save your information. Every time you press **ENTER** or **PF6**, the GROUP EXEC tells you how many user IDs it has processed. Use **PF7** and **PF8** if you have more than one screenful of user IDs.

   b. **Return to the Primary Option Menu.**

      Press **PF3**. If you forget to press **ENTER** or **PF6**, you will remain on this screen and see the message:

      ```
      577E File has been changed; type QQUIT to quit anyway
      ```

      To continue, simply press **PF6**. You will receive a message telling you how many authorized user IDs the EXEC has processed. Press **PF3** again to return to the Primary Option Menu.

```
GRP121            SAVED SYSTEM INFORMATION FOR  <     >        PAGE 1 OF 2
------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.

   SYSTEM DISK address (S-disk) . . . . .:        595
   SYSTEM DISK EXTENSION address (Y-disk):        59E

   USERID to RECEIVE STORAGE DUMPS. . . .:        ________
   RECOVERY MACHINE USERID (required) . .:        ________

   GCS TRACE TABLE SIZE (minimum 4K). . .:        ____16K


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  ...      8  NEXT     9  VERIFY  10  ...    11  ...      12  CURSOR

====>
```

10. To complete the first Saved System Information panel:

    a. **Specify your disk addresses.**

       The default 595 and 59E virtual addresses are already saved and verified. If you must choose different addresses, type over the default values.

       Note: Make sure the new information is correct; the GROUP EXEC does not prevent you from saving invalid information.

    b. **Name an authorized dump receiver.**
       Name an authorized recovery machine.

       GCS requires you to enter a recovery machine user ID. The GROUP EXEC tells you whether that user ID is valid, but it does not prevent you from saving an invalid entry.

    c. **Specify a trace table size.**

       By default, the GROUP EXEC saves a value of 16K. If you decide to save a different amount, simply type the new value over the default value.

    d. **Save your input.**

       If you are satisfied with your choices, press **PF6** or **ENTER** to save the information.

       Warning: If you do not save the information before going on, the system will not accept it as input.

    e. **Go to the second panel.**

       Press **PF8** to continue on the second Saved System Information panel.

```
GRP122            SAVED SYSTEM INFORMATION FOR <      >        PAGE 2 OF 2
--------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.


MAXIMUM NUMBER of VIRTUAL MACHINES (required). .:_____________________________

SYSTEM ID (maximum 130 characters) . . . . . .:_____________________________

_______________________________________________________________________________

_______________________________________________________________________________



--------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  ...      9  ...     10  ...    11  ...     12  CURSOR

====>
```

11. To complete the second Saved System Information panel:

    a. **Specify a MAXIMUM NUMBER.**

       Simply type a value of 1 or more, and press **ENTER**. The GROUP EXEC responds with a message *only* if you enter an invalid value.

    b. **Type in your SYSTEM ID text.**

       This is optional. Move the cursor to the SYSTEM ID space (if it is not already there) and enter your text. The GROUP EXEC does not let you enter any more than 130 characters.

    c. **Save your input and return to the Primary Option Menu.**

       Press **ENTER** to save your information and then press **PF3**.

```
GRP13        AUTOMATIC SAVED SEGMENT LINKS FOR <      >    PAGE 1 OF 1
------------------------------------------------------------------------------


      To ADD,    fill in the blanks with the saved segment names
                 that will be linked automatically during
                 initialization of this virtual machine group.
      To CHANGE, type a new saved segment name over the saved
                 segment name to be changed.
      To DELETE, type blanks on the line.

         Pressing the ENTER key or PF6 key will save the update.

        ________  ________  ________  ________________
        ________  ________  ________  ________________
        ________  ________  ________  ________________
        ________  ________  ________  ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5   REFRESH 6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY  10  ...    11   ...    12  CURSOR

====>
```

12. To complete the Automatic Saved Segment Links panel:

    Note: Do not include CMSVSAM and CMSBAM segments here.

    a. **Type in the segment names.**

       Press **ENTER** or **PF6** after typing each segment name to save your information and advance to the next space at the same time. Every time you press **ENTER** or **PF6**, the EXEC tells you how many segment names it has processed. Use **PF7** and **PF8** if you have more than one screen of names.

    b. **Return to the Primary Option Menu.**

       If you have not saved your input yet, press **ENTER** or **PF6**. Press **PF3** to return to the primary option menu.

13. Once you have provided the necessary input on all panels and returned to the primary option menu for the last time, press **PF3** to exit from the GROUP EXEC.

14. Remove the temporary GCS message file from user storage and erase it on your A-disk:

```
set language langid (delete csi user ■
erase csiume[y] text a ■
```

    langid is your system default national language. If you have not installed a different one, your default is **ameng**.

15. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

16. Invoke the ITASK EXEC with the BUILD GCS parameters:

```
itask build gcs systemname ■
```

systemname must match the name you specified when you issued the GROUP EXEC. If you do not specify a name, the default is GCS.

```
HCPNSD440I The Named Saved System (NSS) systemname
            was successfully defined
            in fileid fileno
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT

DMSUPD181E No update files were found
DMSWHM1907I Assembling systemname
DMSWHM1909I systemname TEXT A created
191 replaces A (191)
5E5 replaces B (5E5)
```

If You Receive an Assembly Error

Note: If you receive an assembly error, you may have to reestablish the minidisk access order:

```
acc 5E5 b ■
vmfsetup 56643089 gcs (all ■ erase 56643089 $setup a ■
```

Then go back and check your configuration file, which ITASK has renamed *systemname* ASSEMBLE. Refer to "Planning for the Group Control System (GCS)" on page 9 for information regarding required fields. Correct the file or go through the GROUP EXEC panels again to recreate the file. (If you recreate the configuration file, you must use the same *systemname*.) After you recreate the configuration file, rename or erase the old *systemname* ASSEMBLE file, then rename the new configuration file from GROUP to ASSEMBLE. Issue the ITASK BUILD GCS *systemname* command to assemble the configuration file and to build and save the nucleus.

End of If You Receive an Assembly Error

```
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
nnnnnnn FILES CHANGED
:
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

These are informational messages. You can ignore them.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The GCS nucleus has been sent to MAINT's virtual reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside GCS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

17. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

18. Copy the text deck, configuration file, ASSEMBLE file, and EXEC from MAINT 191 to MAINT 495:

```
access 495 d ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname text a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname text a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname group a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname group a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname assemble a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname assemble a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

19. Verify that the GCS nucleus is in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss fn  ft       SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 22 on page 88. It has approximately 8,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

The name of the IPL deck is the same as the last entry in the GCS load list. If you have difficulty identifying the IPL deck and need to know the reader file number, refer to the last entry in the GCS load list *systemname* EXEC. (The default name is GCSLOAD EXEC).

20. Order your reader so that the GCS nucleus will be processed first:

```
order rdr fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the GCS nucleus.

21. Query the reader. If it is not class *, spool it as class *:

```
query virtual c ■
RDR 000C CL cl NOCONT NOHOLD EOF READY
    000C nnnnnCLOSED NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
spool c class * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

22. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
MSG FROM MAINT  :  CSIINI134I systemname has nnnnnn
   bytes of available common free storage
HCPNSS440I Named Saved System (NSS) systemname
            was successfully saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT WAS fileno
  RECS nnnn CPY 001 NOHOLD NOKEEP
```

This message indicates that the GCS load map file has been sent to MAINT's virtual printer. If you plan to save or print the GCS load map, record *fileno*.

23. IPL 190:

```
ipl 190 ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: If necessary, purge your reader to free spool space.

Saving and Printing the GCS Load Map

24. The GCS load map has been sent to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME  NAME TYPE   DIST
MAINT    fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

The GCS load map is the file with a blank filename and filetype. It has approximately 2,200 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

```
#cp transfer prt fileno to * rdr ■
access 495 d ■
receive fileno fn ft d ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print fn ft d ■
```

End of Saving and Printing the GCS Load Map

Installing Multiple GCS Systems

25. You can install more than one GCS system, if you defined the additional systems in the system directory.

    To install each additional system:

    a. Reestablish the minidisk access order:

    ```
    access 5E5 b ■
    vmfsetup 56643089 gcs (all ■
    erase 56643089 $setup a ■
    ```

    b. Set up the GROUP EXEC messages in storage (full-screen display devices only):

    ```
    copyfile csimes[y] [text|txtnnnnnn] fm csiume[y]text a ■
    ```

    The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a UM*nnnnn* field, where *nnnnn* is the PTF number.

    Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*. You will have one with a filetype of TEXT.

    ```
    set language langid (add csi user ■
    ```

    c. Create a configuration file for each nucleus.

    - If you are using a full-screen display device, issue:

      ```
      group systemname ■
      ```

      systemname is a unique filename. Then complete the GCS panels to define the system parameters.

    - If you are **not** using a full-screen display device, refer to the *VM/XA SP Group Control System Command and Macro Reference*. Use the build macros to create the configuration file.

    d. Remove the temporary GCS message file from user storage and erase it from your A-disk (full-screen display devices only):

    ```
    set language langid (delete csi user ■
    erase csiume[y] text a ■
    ```

    e. For each new GCS system, add a DEFSYS statement to the SAMPNSS EXEC. For information on the DEFSYS command, see the *VM/XA SP CP Command Reference*.

f. For each new GCS system, add an override section at the end of the product parameter file. This example shows an override section for a GCS system called **NEWGCS**:

```
:NEWGCS. GCS
:BLD. REPLACE
 NEWGCS VMFBDNUC BUILD1
:END.
```

You must also add the name of the new GCS system to the :OVERLST. tag at the top of the product parameter file, for example:

```
:OVERLST. CORCP CORCMS CORGCS NEWGCS
```

g. For each GCS system, assemble the configuration file, build and save the new GCS nucleus, and copy the text deck, configuration file, ASSEMBLE file, and EXEC to the 495 disk:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■

itask build gcs systemname ■
access 495 d ■
copy systemname text a = = d (replace ■
erase systemname text a ■
copy systemname group a = = d (replace ■
erase systemname group a ■
copy systemname assemble a = = d (replace ■
erase systemname assemble a ■
ipl 00c clear ■
ipl cms ■
```

Before you build another GCS nucleus, save or print the GCS load map, as indicated above. Remember to give each load map file a different name.

End of Installing Multiple GCS Systems

26. Verify that all your GCS segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss GCS         NSS      MAINT
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss systemname  NSS      MAINT
⋮
```

27. When you complete your GCS installation, go to "Step 27. Save CMS" on page 91.

Step 27. Save CMS

In this step, you will save CMS as a named saved system in 370 mode and 370-XA mode, at default addresses.

This step takes approximately 10 minutes.

Note: For general information about CMS as a named saved system, see *VM/XA SP Planning and Administration.*

1. Make sure:
   - You are logged on as MAINT.
   - You have finished loading all files to the 190 and 19E minidisks. After this step, any change to files on these disks requires resaving CMS.
   - You have 16 megabytes of virtual storage. To determine your storage size, issue QUERY VIRTUAL STORAGE. If you have less than 16 megabytes, issue DEFINE STORAGE 16M.
   - CMS is running. If not, load CMS by issuing IPL 190 CLEAR. Wait for the CMS version identification to appear on the screen. Then press **ENTER**.
2. Invoke the SAMPNSS EXEC:

```
sampnss cms cmsxa ■
```

The SAMPNSS EXEC issues the DEFSYS command.

Note: If you use "CMS" (the default name) for the System/370 name, users will receive the new level of CMS whenever they issue the command IPL CMS.

```
HCPNSD440I Named Saved System (NSS) CMS was
     successfully defined in fileid fileno

HCPNSD440I Named Saved System (NSS) CMSXA was
     successfully defined in fileid fileno
```

The SAMPNSS EXEC first issues the DEFSYS command to define a skeleton system data file for CMS and for CMSXA. These messages tell you the DEFSYS commands were successful.

The names shown in the SAMPNSS command are the default names for the System/370 (CMS) and for the 370-XA (CMSXA) systems. Of course, you may name these systems whatever you wish; but if you change the names you must either edit the SAMPNSS EXEC to use your names or define your systems manually.

Note: The names are positional on the command (the System/370 name followed by the 370-XA name).

```
OWNERID  FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMS      NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The SAMPNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

3. Issue the QUERY NSS ALL MAP command to make sure CMS is defined properly. Check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Set your machine mode to 370, load CMS (IPL 190), and save the CMS system:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

If your machine is already in 370 mode, you will not get a response.

```
ipl 190 clear parm savesys cms ■
HCPNSD440I Named Saved System (NSS) CMS was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMS.

```
DMSWSP327I The installation saved segment
   could not be loaded
```

Disregard the message about not loading the saved segment; CMS has been successfully saved as a named saved system.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

5. Set your machine mode to XA, load CMS (IPL 190), and save the CMSXA system:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

If your machine is already in 370-XA mode, you do not get a response.

```
ipl 190 clear parm savesys cmsxa ■
HCPNSD440I Named Saved System (NSS) CMSXA was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMSXA.

```
DMSWSP327I The installation saved segment
   could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Disregard the message about not loading the saved segment; CMSXA has been successfully saved as a named saved system.

Press **ENTER** to initialize CMS.

6. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the CMS (or CMSXA) named saved system. The example below shows loading CMS in System/370 mode:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
define storage 2m ■
STORAGE = 2M
Storage cleared - system reset
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If your machine is already in 370 mode, you will not get a response.

Defining storage causes a system reset.

After receiving the CMS version identification, press **ENTER** to complete CMS or CMSXA initialization.

7. Check the CMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Check whether the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Redefine your storage before you continue with the following steps:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 28. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments

If you plan to use the CMS/DOS environment or the VSAM product, you must install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS saved segments.

If you do not want to install these saved segments, go to "Step 29. Create the CMSINST and HELP Saved Segments" on page 103.

In this step, you will:

- Install CMSDOS and CMSBAM in separate saved segments
- Create a CMSVSAM segment and install the VSAM product tape
- Create a CMSAMS segment and install Access Method Services.

This step takes approximately 1 hour.

Warning: The CMSDOS and CMSBAM segments must be installed before you install CMSVSAM and CMSAMS.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. IPL 190. This is the CMS system disk.

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
```

`VM/XA CMS 5.6 mm/dd/yy hh:mm`

`** DO NOT press ENTER! **`

Your own installation header appears if you have changed it.

```
access (noprof ■
Ready; T=n.nn/n.nn hh:mm:ss
```

This command suppresses execution of MAINT's PROFILE EXEC.

```
access 193 e ■
access 5E5 b ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Access the disks with installation and service EXECs, and the DOSGEN EXEC.

```
set emsg on ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You want to see any and all error messages during execution of the installation EXEC.

2. Define a segment into which the DOSGEN EXEC can load CMSDOS:

```
defseg dosinst 900-90f sr ■
HCPNSD440I Saved segment DOSINST was
           successfully defined in fileid
           fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

This command defines a 1M segment for DOSINST starting at 900000. You may place this segment anywhere below the segment spaces defined for CMSDOS, CMSBAM, CMSVSAM, and CMSAMS.

3. Invoke the DOSGEN EXEC with a load address and the name DOSINST. The load address used for DOSINST in this example is 900000.

```
dosgen 900000 dosinst ■
HCPNSS440I Saved segment DOSINST was
            successfully saved in fileid
            fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWGN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the DOSINST segment. Error messages for the DOSGEN EXEC are listed on page 97.

4. IPL CMS:

```
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Define CMSDOS, CMSBAM, CMSAMS, and CMSVSAM:

```
set sysname cmsdos dosinst ■
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsdos ■
HCPNSD440I Saved segment CMSDOS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss DOSBAM   DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSDOS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsbam ■
HCPNSD440I Saved segment CMSBAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSBAM   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsams ■
HCPNSD440I Saved segment CMSAMS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSAMS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
sampnss cmsvsam ■
HCPNSD440I Saved segment CMSVSAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSVSAM  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Invoke the DOSGEN EXEC with a load address and the name CMSDOS. The load address set up by SAMPNSS is B00000 for the name CMSDOS.

```
access 193 e ■
Ready; T=n.nn/n.nn hh:mm:ss
dosgen b00000 cmsdos ■
HCPNSS440I Saved segment CMSDOS was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSGEN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

Access the disks with installation and service EXECs.

The load address and name are those recommended for the CMSDOS segment. Error messages for the DOSGEN EXEC are listed on page 97.

7. To save the load map, rename it and copy it to the alternate LOCAL1 minidisk (395):

```
access 395 d ■
DMSACC724I 395 replaces D(395)
Ready; T=n.nn/n.nn hh:mm:ss
copy load map a cmsdos segmap d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Error Messages from DOSGEN

If DOSGEN detects an error in the address that you specified:

```
DMSGEN095E INVALID ADDRESS
```

If DOSGEN cannot find a read/write accessed A-disk:

```
DMSGEN006E NO READ/WRITE A-DISK ACCESSED
```

If DOSGEN finds unresolved external references while loading the text files:

```
DMSGEN111E DOSGEN FAILED DUE TO LOAD ERRORS
```

If DOSGEN detects an error while assigning the storage key or saving the segment:

```
DMSGEN412S DOSGEN FAILED DUE TO SETKEY ERRORS

DMSGEN141S DOSGEN FAILED DUE TO SAVESYS ERRORS
```

8. Replace the name CMSBAM in the CMS SYSNAME table with any name that is **not** a name previously used as a segment name:

```
set sysname cmsbam sysname ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The SET SYSNAME command enters an alternate name for CMSBAM in the SYSNAME table. Choose a name that was **not** used previously as a segment name. This command does not actually change the saved segment's name, but keeps CP from finding and IPLing the segment at the wrong time.

9. Place your CMS virtual machine in a CMS/DOS environment, then invoke SAMGEN to load the CMSBAM segment. You must give the EXEC an address at which to load the CMSBAM segment; this address also must match the address in the skeleton CMSBAM segment that you defined beforehand.

```
set dos on ■
```

This command places your virtual machine in a CMS/DOS environment.

```
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
             at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
samgen ■
DMSSGN363R ENTER LOCATION WHERE CMSBAM
WILL BE LOADED AND SAVED:

b10000 ■
DMSSGN364I FETCHING CMSBAM...
DMSFET710I PHASE DMSVBM ENTRY POINT AT LOCATION B100C0.
DMSSGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsbam ■

HCPNSS440I Saved segment CMSBAM was successfully
            saved in fileid fileno
DMSSGN365I SYSTEM CMSBAM SAVED
Ready; T=n.nn/n.nn hh:mm:ss
```

You may receive these informational messages. *sysname* is the name you chose in substep 8.

The messages indicate that the segment has been loaded and saved.

10. Set DOS off:

```
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

11. Define and format a minidisk for VSAM files.

    a. Map your USER DIRECT and examine the map to find a space for the new minidisk. Look for a gap of appropriate size and record the starting cylinder of that gap. The number of cylinders needed depends on the DASD type:

Device	Cylinders Needed
3350	15 cylinders
3375	22 cylinders
3380	14 cylinders
3390	14 cylinders

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
diskmap user direct ■
File USER DISKMAP A has been created.
Ready; T=n.nn/n.nn hh:mm:ss
xedit user diskmap ■
locate/GAP ■
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Refer to the directory map to find a gap. Repeat the **locate** command as many times as necessary. When you find a suitable gap, note the starting cylinder and DASD label the cylinders reside on and leave the file.

b. XEDIT your directory and add an entry for the new minidisk:

```
xedit user direct ■
locate/USER MAINT ■
locate/MDISK ■
input ■
MDISK vdevno devtype startcyl cyls label MW ALL ■
■
file ■
```

Locate the MAINT minidisk definitions.

vdevno, *devtype*, and *label* are the address, device type, and label of the DASD volume where the new minidisk is located. *startcyl* is the starting cylinder of the gap where the new minidisk is defined. *cyls* is the number of cylinders needed for the minidisk.

Press **ENTER** to return to the command line.

c. Map the directory again and examine the map to make sure that the new minidisk was created in the place you specified and that it does not overlap any existing minidisks:

```
diskmap user direct ■
```

d. Issue DIRECTXA to update the directory and bring it online:

```
directxa user ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ONLINE
Ready; T=n.nn/n.nn hh:mm:ss
```

e. Link to the new minidisk:

```
link * vdevno vdevno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

f. Format the new minidisk:

```
format vdevno v (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk V(vdevno).
            Do you wish to continue?
            Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
label ■
DMSFOR733I Formatting disk V
DMSFOR732I nn cylinders formatted on V(vdevno)
Ready; T=n.nn/n.nn hh:mm:ss
```

g. Access the new minidisk as A:

```
access vdevno a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

12. Set DOS on:

```
set dos on ■
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
              at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
```

You may receive these informational messages. *sysname* is the name you chose in substep 8 on page 98.

13. Mount the VSAM product tape on a tape drive attached to MAINT at virtual address 181. (If you do not have the proper class authority to attach a tape, the system operator may have to do it for you.)

14. Invoke VSAMGEN EXEC:

```
vsamgen ■
SELECT ONE OF THE FOLLOWING FUNCTIONS BY ENTERING THE NUMBER:

 1. INSTALL AMS            (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 2. INSTALL VSAM           (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 3. INSTALL VSAM AND AMS   (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 4. BUILD  AMS             (BUILD DOSLIB, CREATE SEGMENT)
 5. BUILD  VSAM            (BUILD DOSLIB, CREATE SEGMENT)
 6. BUILD  VSAM AND AMS    (BUILD DOSLIB, CREATE SEGMENT)
 7. RESTART AMS            (CREATE SEGMENT)
 8. RESTART VSAM           (CREATE SEGMENT)
 9. RESTART VSAM AND AMS   (CREATE SEGMENT)
10. QUIT                   (EXIT VSAMGEN EXECUTION)

      ENTER RESPONSE...
```

Choosing option 3 tells the EXEC to create both CMSVSAM and CMSAMS segments as new segments.

If this is the initial installation of VSAM, select function **1**, **2**, or **3**.

If the text files have already been created from the VSAM product tape and are currently on the accessed A-disk, select function **4**, **5**, or **6**.

If the DOSLIBs currently reside on an accessed disk, select function **7**, **8**, or **9**. In this case, it is not necessary to have the text files available.

```
3 ■

DMSVGN365R ONE OR MORE OF THE TEXT FILES LISTED IN THE CMSVSAM EXEC
            ARE MISSING.  THE VSAM PP PID TAPE SHOULD BE ON TAPE DRIVE 181
            TO RESTORE THE FILES. ENTER:
            'GO' IF TAPE DRIVE IS READY TO LOAD FILES,
            'QUIT' TO STOP GENERATION PROCESS.
go ■
```

Messages from VSAMGEN

While VSAMGEN is processing, you may receive error and information messages. These messages are self-explanatory. Messages labeled 2101I are information messages from the linkage editor and may be ignored.

Respond to the EXEC messages as they appear:

```
DMSVGN362I LINK-EDITING CMSVSAM ...
DMSVGN363I CMSVSAM DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSVSAM SHARED WILL BE
          LOADED AND SAVED:

ba0000 ■

DMSVGN363R ENTER LOCATION WHERE CMSVSAM NONSHARED WILL BE
          LOADED AND SAVED:
a30000 ■

DMSVGN364I FETCHING CMSVSAM ...
DMSFET710I PHASE DMSVVS ENTRY POINT AT LOCATION nnnnnn
DSMVGN371R CMSVSAM IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:

cmsvsam ■

DMSVGN365I SYSTEM CMSVSAM SAVED.

DMSVGN368R ERASE CMSVSAM DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■

DMSVGN362I LINK-EDITING CMSAMS ...
DMSVGN363I CMSAMS DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSAMS SHARED WILL BE
          LOADED AND SAVED:
```

This name is the default name with which the segment was defined.

```
b40000 ■                                          This is the recommended location.
DMSVGN363R ENTER LOCATION WHERE CMSAMS NONSHARED WILL BE
         LOADED AND SAVED:
a00000 ■
DMSVGN364I FETCHING CMSAMS ...
DMSFET710I PHASE DMSVAS ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAN ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAX ENTRY POINT AT LOCATION nnnnnn
DMSVGN371R CMSAMS IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
       'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsams ■                                          This name is the default name used for defining
                                                  the segment.
DMSVGN365I SYSTEM CMSAMS SAVED.

DMSVGN368R ERASE CMSAMS DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
Ready; T=n.nn/n.nn hh:mm:ss
```

15. You may now purge the DOSINST named saved segment:

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
purge nss fileno ■                                fileno is the file identifier for DOSINST.
No files purged
0001 file pending purged
Ready; T=n.nn/n.nn hh:mm:ss
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 29. Create the CMSINST and HELP Saved Segments

In this step, you will:

- Define a segment to contain EXECs and System Product Editor macro instructions
- Use the DCSSGEN command to load and save the CMSINST segment
- Save the HELP file directory in a saved segment.

Notes:

1. You must have the NAMESAVE HELP statement in your user directory in order to save the HELP file directory information in a saved segment.

   See "Step 21. Update the User Directory" on page 71.

2. If the HELP segment is defined as a member of a segment space, you must have a NAMESAVE statement (for example, NAMESAVE INSTHELP) in the directory entry for the user who invokes SAVEFD. SAVEFD specifies the name of the segment space.

This step takes approximately 20 minutes.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. Log on as MAINT (unless you are continuing from the previous step).
2. Check your virtual storage. If it is less than 16M, issue the following:

```
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 0016M
```

3. IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
```

Press **ENTER** to complete the CMS initialization.

```
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Use the SAMPNSS EXEC to define a segment for CMSINST:

   Note: If you have increased the number of entries in the CMS load list, be sure that the DEFSEG entry for CMSINST in the SAMPNSS EXEC has enough pages defined to accomodate the increased size.

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
           defined in fileid fileno
```

```
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The EXEC issues a QUERY command that displays information about the segments defined for named saved systems.

5. Define the HELP segment:

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Save both segments:

```
saveseg cmsinst ■
HCPNSS440I Saved segment CMSINST was
            successfully saved in fileid
            fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg help ■
HCPNSS440I Saved segment HELP was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Each member saved segment defined by SAMPNSS must be saved separately.

7. Redefine each segment to create the skeleton segments:

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If you do not want to use the sample load list, CMSINST EXECLIST, for the DCSSGEN command, use the System Product Editor (XEDIT) to create your own load list:

```
xedit instlist file a ■
input ■
```

```
* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S ■
```

This is a sample. Enter the load list entries you need.

Only one comment line per list is allowed; it must be the first line. For more information on creating this list, see "DCSSGEN Command" on page 580.

The remainder of the lines contain entries for frequently used EXECs and Editor macros; multiple users can share the same executing copy of these EXECs and macros.

```
■
file ■
```

Press **ENTER** to return to the command line.

9. The load list must have a fixed record length in order for you to use the DCSSGEN command. If your load list has variable length records, change it to a fixed length file by issuing:

```
copyfile fn ft fm = = = (recfm f lrecl 80 replace ■
```

10. Invoke the DCSSGEN command:

```
access 193 e ■
dcssgen fn ft fm cmsinst ■
```

fn ft fm is the file ID of the file that contains the list of EXECs and Editor macros to be loaded into the segment, either CMSINST EXECLIST *fm* or your own file (for example, INSTLIST FILE A).

CMSINST is the name of the segment you want to create. It is also the default name of the installation saved segment in the DMSNGP profile. If you changed the name in the DMSNGP profile, or specified a different name in response to the installation questions (see page 50), use that name instead. If you do not specify a segment name, the DCSSGEN command defaults to CMSINST.

```
HCPNSS440I Saved segment CMSINST was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

The DCSSGEN command creates a load map file, CMSINST DCSSMAP A. For the sample load list created in substep 8 on page 105, the load map would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
    15:41:59 10/22/85 copy of RDRLIST  EXEC     S loaded as MAIL     EXEC
    EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
    15:41:58 10/22/85 copy of FILELIST EXEC     S loaded as FILELIST EXEC
    EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
     7:30:18 11/26/85 copy of SYSPROF  EXEC     S loaded as SYSPROF  EXEC
    EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
     8:47:55 12/18/84 copy of PARSE    XEDIT    S loaded as PARSE    XEDIT
    EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
    15:40:32 10/22/85 copy of DISCARD  EXEC     S loaded as DISCARD  EXEC
    EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
    11:03:53 10/24/85 copy of NOTE     EXEC     S loaded as NOTE     EXEC
    EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
    15:41:55 10/22/85 copy of PROFNOTE XEDIT    S loaded as PROFNOTE XEDIT
    EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
    15:41:05 10/22/85 copy of ALL      XEDIT    S loaded as ALL      XEDIT
    EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
    *** End of Source List ***
    CMSINST built at 15:56:34 on 12/02/85
```

Note: When you built your CMS nucleus, if you indicated in the DMSNGP file (USEINST=YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs CMS. If you previously indicated that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild CMS. (To find the latest version of the DMSNGP file, issue SETUP to access all the minidisks where it might be; then issue FILELIST DMSNGP ASSEMBLE *. All the copies of DMSNGP ASSEMBLE will be listed, with the date when each one was last modified.)

Messages from DCSSGEN

While DCSSGEN is processing, you may receive error or warning messages that indicate specific conditions. If errors were encountered, after processing is complete you receive the following prompt:

```
DMSEXG298R An error has been detected while building the DCSS.
           Do you still want the DCSS saved?
           Enter 1 (YES) or 0 (NO).
```

Enter **1** to disregard the errors and save the segment, or enter **0** to not save the segment. If you do not save the segment, you receive the message:

DMSEXG288I *segname* not saved

If DCSSGEN encounters an error while saving the segment, you receive the message:

DMSEXG288E *dcssname* not saved

If your virtual machine is not large enough to contain the segment (you need 16M), you receive the message:

```
DMSEXG284E The DCSS is not completely inside the virtual machine
```

To correct this situation, increase the size of your virtual machine, re-IPL CMS, and reissue the DCSSGEN command.

11. Define your virtual storage **less** than the address at which the HELP segment is to be loaded. For example, if the HELP segment is defined at X'C50000' (the default address in the SAMPNSS EXEC) define your storage as 12M.

```
define storage 12m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 12M
```

12. Re-IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

13. Issue the following commands to initialize and save the segment:

    (For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*.)

```
savefd init vaddr label help ■
Ready; T=n.nn/n.nn hh:mm:ss
savefd save vaddr label help ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is 19D for mixed-case American English and 19C for uppercase American English.

label is the CMS label assigned to the disk. (The labels in the sample directory are MNT19D and MNT19C.)

14. Verify that the CMSINST and CMSHELP segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss INSTHELP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss CMSINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss HELP        NSS      MAINT
```

The system data file should now have class A (rather than S) and have one user (MAINT).

15. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 30. Back Up the Named Saved System Files

In this step, you will back up all the named saved systems, including CMS, on tape.

This step takes approximately 10 minutes.

1. Mount a scratch tape with ring on an available 3420, 3422, or 3430 tape drive, or insert a cartridge with the tab protector set to off in an available 3480 tape unit. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT.
   - The tape must be suitable for a 3420, 3422, 3430, or 3480 tape unit.
2. Issue the SPTAPE command to dump the CMS system data file to tape:

   ```
   sptape dump rdevno nss all run ■
   ```

   Substitute the real device number of the tape drive for the value *rdevno*. The operand RUN specifies that the SPTAPE rewinds and unloads the tape after the operation.

   ```
   DUMPING rdevno NSS *NSS      nnnn A
   ⋮
   SPTAPE DUMP FUNCTION ON DRIVE rdevno
      COMPLETE
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   The messages from SPTAPE tell you that the file is being dumped to tape. The CMS ready message may occur between the messages.

3. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE command to restore the CMS system data file. For more information about the SPTAPE command, see *VM/XA SP Real System Operation.*

To restore your CMS named saved system:

1. Log on as MAINT.
2. Mount the backup tape on a tape drive.
3. Issue:

   ```
   sptape load rdevno nss all run ■
   LOADING rdevno NSS *NSS nnnn A    NOW nnnn
   ⋮
   LOADING rdevno NSS *NSS nnnn A    NOW nnnn
   SPTAPE LOAD FUNCTION ON DRIVE rdevno COMPLETE
   ```

4. IPL CMS:

   ```
   ipl cms ■
   VM/XA CMS 5.6 mm/dd/yy hh:mm
   ■
   SYNONYM SYN
   CP TERM MODE VM
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

Step 31. Store a Backup Copy of CP on Tape

In this step, you will use DDRXA to store a backup copy of CP on tape.

The time to perform this step varies based on the device and contention for the device's channel. It may take up to 30 minutes.

1. Mount a scratch tape (with ring) on a tape drive. Follow machine operations manuals to mount the tape.
2. Attach the tape drive to MAINT at virtual device number 0181:

```
attach rdevno * 181 ■
TAPE     0181 ATTACHED
Ready; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the real device (*rdevno*) to MAINT's virtual machine at virtual device number 0181.

3. Load the DDRXA utility to tape. If your tape drive can operate at a density of 6250 BPI, enter the commands under 3a. Otherwise, enter the commands under 3b.
   a. If your tape drive can operate at a density of 6250 BPI, enter these commands:

```
tape modeset (tapl den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Set the tape density to 6250 BPI. If you receive message DMSPAR622E, reissue the TAPE MODESET command.

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl (den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

Go to substep 4.

   b. If your tape drive cannot operate at a density of 6250 BPI, enter these commands:

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

4. Change MAINT's virtual processor configuration to 370-XA architecture:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

Setting the virtual machine to 370-XA architecture causes a reset as if you entered SYSTEM CLEAR. If your machine is already in 370-XA mode, you will not get a response.

```
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Rewind the scratch tape on virtual device number 0181:

```
rewind 181 ■
Rewind complete
```

6. Load (IPL) the tape and answer the prompts from DDRXA:

```
ipl 181 clear ■
```

Wait a few moments for DDRXA to prompt you. If a prompt does not appear, press the **ENTER** key.

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:
```

```
sysprint cons ■
ENTER:
```

This first control statement tells DDRXA that you want program messages sent to your console.

```
input vdevno devtype xasres ■
ENTER:
```

The second control statement is the input control statement. *vdevno* is the virtual address of the system residence volume. (The virtual address in the sample HCPSYS ASSEMBLE file and USER DIRECT is 123.) *devtype* is the DASD type of this volume.

```
output 181 devtype (mode 6250 compact ■
ENTER:
```

This control statement specifies the device to which you are dumping the system.

If your tape drive is not operating at 6250 BPI density, omit the **mode 6250 compact** option on this command.

```
dump cpvol ■
```

The statement DUMP CPVOL dumps cylinder 0 and all PERM and DRCT DASD space from the system residence volume to the tape.

```
DUMPING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn    0000   nnnn
END OF DUMP
BYTES IN nnnnnnnnnn BYTES OUT nnnnnnnnnn
TRACKS NOT COMPACTED ON TAPE - nnnnnnnnnn
```

These are informational messages. GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

```
ENTER:
■
END OF JOB
```

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

You now have a loadable tape of the system. If you need to use this backup copy to restore CP, perform these steps:

1. Mount the backup tape on a tape drive.
2. Set your virtual machine to 370-XA architecture, define your virtual storage as 16 megabytes, and IPL CMSXA:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
define storage 16m ■

STORAGE = 16M
Storage cleared - system reset
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If your machine is already in 370-XA mode, you will not get a response.

3. Perform a hardware load (IPL) of the tape device.
4. Use DDRXA to restore the system to disk:

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:

sysprint cons ■
ENTER:

input rdevno devtype ■
ENTER:

output rdevno devtype xasres ■
ENTER:

restore all ■

RESTORING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO XASRES
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn     0000   nnnn
⋮
END OF RESTORE
BYTES RESTORED nnnnnnnnnn

ENTER:
■
END OF JOB
```

This first control statement tells DDRXA that you want program messages sent to your console.

The second control statement is the input control statement. Identify the device number (*rdevno*) and tape device type (3420, 3422, 3430, or 3480) where the backup tape is mounted.

This control statement specifies the DASD device to which you are restoring the system (XASRES).

The RESTORE ALL statement tells DDRXA to restore the whole tape to the output device.

Informational messages: GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

Chapter 3. Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)

If you have a first-level VM/XA MA, VM/XA SF, or VM/XA SP system, you can install VM/XA System Product as a second-level guest of that system. (For good performance, your first-level system should have at least 16 megabytes of storage.) Otherwise, you must install VM/XA System Product as a first-level system. See Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" on page 13 or Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215.

If your first-level system is VM/XA SP, the first-level user ID you use to install your new system must have the privilege classes that are specified on the SERV parameter of the SYSFCN macro in your first-level system's HCPSYS ASSEMBLE file. If this user ID does not have those privilege classes, you will not be able to run the Device Support Facilities.

Step 1. Load the Device Support Facilities

In this step, you will load the Device Support Facilities (DSF) and use it to initialize and inspect the DASD for defective tracks. For more information about the Device Support Facilities, see the *Device Support Facilities User's Guide and Reference*, SC35-0033.

This step takes approximately 20 to 40 minutes for each DASD volume you inspect, depending on the type of DASD.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Before you begin, make sure that you have read the information about these steps in Chapter 1, "Introduction" on page 3.
2. Ready the DASD volumes you plan to use. Follow the operation manual for your own hardware.

   If the packs you plan to use for your installation are not already ICKDSF-initialized, you must initialize them before proceeding.

   Warning: No volumes with labels XASERV, XAP001, or XAP002 should be attached to your system. Any such volumes will be brought on line when you IPL the starter system in Step 3. If necessary, you can use the Device Support Facilities to relabel your packs. Issue REFORMAT UNIT(*vdevno*) VERIFY(*oldlabel*) VOLID(*newlabel*) when DSF prompts you with `ENTER INPUT/COMMAND`.

   Warning: The addresses of XASRES, XASERV (if used), XAP001 (if used), and XAP002 (if used) must match the addresses in the HCPRIO ASSEMBLE file. If the addresses do not match, you will not be able to IPL your new system in Step 20. If your addresses are not defined in the sample HCPRIO ASSEMBLE file for your DASD type, you will have to tailor it in Step 17.
3. Attach a tape drive to your user ID. If possible, attach the tape drive at the default virtual address, 0181. (This is the address used throughout this procedure; if your tape drive has a different virtual address, make a note of that address on Table 1 on page 7.) Have the starter system tape mounted on the tape drive.

4. Issue:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

If your machine is already in 370-XA mode, you will not get a response.

5. Issue:

```
ipl vdev clear ■
```

Use the address of the tape drive on which the starter system is loaded.

Pause for a short while to allow the IPL to complete, then press the **ENTER** key to create an interrupt.

```
■
ICK005E DEFINE INPUT DEVICE, REPLY
'DDD,CUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:
console ■
ICK006E DEFINE OUTPUT DEVICE, REPLY
'DDDD,CUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:
console ■
ICKDSF - SA DEVICE SUPPORT FACILITIES 9.0 TIME:hh:mm:ss
   mm/dd/yy    PAGE 1
ENTER INPUT/COMMAND:
```

This message tells you that the Device Support Facilities are loaded and ready. **You MUST use the Device Support Facilities to format your XASRES volume unless it is already formatted.** If XASRES is already formatted, press **PA1** to exit the Device Support Facilities program and go on to Step 2.

```
init unit(vdevno) devtype(33xx) novfy val volid(volid) ■
```

vdevno is the address of the DASD volume you want to inspect. 33*xx* is the device type: 3350, 3375, 3380, or 3390. *volid* is the volume identifier, in this case, XASRES.

This command requests medial initialization of the volume. For maximal or minimal initialization, see the *Device Support Facilities User's Guide and Reference*.

```
ICK00700I DEVICE INFORMATION vdevno IS CURRENTLY AS FOLLOWS:
          PHYSICAL DEVICE = 33xx
          STORAGE CONTROLLER = xx
          STORAGE CONTROL DESCRIPTOR = xxxx
          DEVICE DESCRIPTOR = xx
ICK003D REPLY U TO ALTER VOLUME vdevno CONTENTS, ELSE T
ENTER INPUT/COMMAND
u ■
⋮
ENTER INPUT/COMMAND:
```

The system takes 20 to 40 minutes to inspect and reformat the DASD volume. After the DASD volume has been reformatted, you receive a series of ICK messages that describe the status of the device being formatted.

6. If you want to inspect or format any more DASD volumes, repeat the INIT UNIT command. Otherwise, press **PA1** to go on to Step 2.

Step 2. Restore the VM/XA System Product Starter System to Disk

In this step, you will use the DDRXA utility to restore the starter system to disk.

This step takes approximately 15 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Issue:

   `ipl` *vdev* `clear` ■ — Use the address of the tape drive on which the starter system is loaded.

2. Establish an interactive dialog with DDRXA:

   Notes:

   a. When you run DDRXA on a real processor, and you have a nonconsole device at real device number 0009 or 001F, make sure the device is not operational. Otherwise, results are unpredictable. (DDRXA initially assumes that your console has device number 0009 or 001F.)

   b. Also, to prevent inadvertent access to DDRXA, **turn off** all displays other than the one you will be using for DDRXA, so that no one will press the **ENTER** key on another display and interrupt DDRXA while it is waiting for input.

   As soon as you load it into storage, DDRXA begins executing and sends you the following initial prompt:

   ```
   VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
   ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
   ENTER:
   ```

   Pause for a short while to allow the IPL to complete, then press the **ENTER** key to create an interrupt.

3. Answer the prompting messages from the DDRXA utility:

   Note: The device types for which the following steps are valid include 3350, 3375, 3380, and 3390.

   ```
   VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
   ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
   ENTER:
   ```

   `sysprint cons` ■
   `ENTER:`

   This first control statement tells DDRXA that you want program messages sent to your display.

   `input 181` *devtype* `(skip 1` ■
   `ENTER:`

   The second control statement is the input control statement. Identify the virtual device address (181) and tape device type (3420, 3422, 3430, or 3480) where the starter system tape is mounted. The option **skip 1** tells DDRXA to skip over the tape mark placed on the tape just before the starter system.

```
output vdevno devtype scratch ■
ENTER:
```

This control statement specifies the virtual address and device type of the system residence volume to which you are restoring the system (XASRES). Specify the device number (*vdevno*) and device type (*devtype*). Valid *devtype* values are:

3350
3375
3380
3390

The word **scratch** causes DDRXA not to check the label on the volume.

```
restore all ■
```

The RESTORE ALL statement tells DDRXA to restore the whole tape to the output device.

```
HCPDDR725D SOURCE DASD DEVICE WAS (IS) LARGER
              THAN OUTPUT DEVICE
           DO YOU WISH TO CONTINUE?
           RESPOND YES OR NO:
yes ■
```

Depending on the DASD model you are using, you may or may not see this prompt. If you do see it, respond **yes**. The starter system is designed to fit on all supported DASD models.

```
RESTORING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      cccc     cccc    cccc   cccc
END OF RESTORE
BYTES RESTORED nnnnnnnnnn

ENTER:
```

Informational messages: GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

```
■
END OF JOB
```

Press **ENTER** to end the program.

4. Go to CP READ:

PA1

This entry puts you in first-level CP. Press **PA1** whenever you need to issue a first-level CP command.

Step 3. Load the VM/XA System Product Starter System and Define Devices

In this step, you will:

- Load the VM/XA System Product starter system
- Define the devices necessary to perform the system generation.

Contention for service by the devices on shared control units may result in this step taking a considerably longer time than it would when you are installing a first-level system. The step could take as little as 15 minutes, but may take considerably longer.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

Local Terminals Only

- If you are using **local terminals** (whose controller is connected to the host computer through a direct channel) to do this installation, issue the following commands:

```
terminal conmode 3270 ■
```

This command provides full-screen System Product Editor (XEDIT) support on your second-level terminals. The display units must be local non-SNA devices. If you log off or disconnect and then log on again, you must reissue this command.

Note: If you receive the message `DMKCFT006E Invalid device type`–, you do not have a local terminal.

End of Local Terminals Only

1. Your first-level system should have at least 16 megabytes of storage:

```
define storage 16m ■
```

RESTART POINT

This is the restart point if you have a failure while building the Starter System at second level.

Re-IPLing Your Second-Level System

If you **log off** your **first-level** user ID during this step or any of the remaining steps in this chapter, you must reissue the following sequence of first-level commands to reestablish the second-level environment before you can re-IPL your second-level system.

set machine xa ■	Your first-level system must be in XA mode.
define storage 16M ■	Your first-level system should have at least 16 megabytes of storage.
term conmode 3270 ■	This command is for local terminals only, and provides full-screen XEDIT support on your second-level terminals.

You must also make sure that your I/O devices are attached to your second-level system and varied on.

2. IPL the XASRES volume:

`ipl` *vdevno* `clear` ■	*vdevno* is the **first** level **virtual** address of the XASRES volume.
■	Pause for a short while to allow the IPL to complete, then press the **ENTER** key to create an interrupt.

Note: If the missing interrupt handler (MIH) is turned on, the following message appears repeatedly:

`5461 Interruption cleared; device` *devtype*`, CSW` *csw*`, userid` *userid*

This is not an indication of a problem; the system is operating correctly.

3. Define your Starter System configuration by answering the following series of prompts.

   The Starter System provides default device addresses and device types. To accept the Starter System default for a device prompt, just press the **ENTER** key.

   IBM recommends that you have a second tape drive to send dumps to when the system is operational.

   You do not need a printer, card reader, punch, or additional graphic device to generate the system. However, your system provides a virtual equivalent for each of these devices. If you do not have these devices, press **ENTER** to accept the Starter System defaults.

   If you do not want to accept a Starter System default, enter the address (*vdev*) or device type (*devtype*). If you supply an address for a device, use the **first** level **virtual** address. At second level, this address functions as a "real" address. Refer to Table 1 on page 7 to help you complete this section.

   Whether you accept the Starter System default addresses or define your own, make sure that your second-level "real" addresses match your first-level virtual addresses.

4. Use Table 1 on page 7 to answer the questions that the VM/XA System Product starter system asks you. The following are the starter system's informational messages and questions:

   Note: If you do not enter values, the system will use its default values. The system default is shown in the right-hand column. However, if the system defaults are not in your IOCDS, the system will not be able to use those devices.

```
DO YOU WISH TO REDEFINE YOUR SYSTEM (YES/NO):
```

```
yes ■
```

If you have not previously defined your system by executing this substep, answer **yes**. If you answer **no**, your starter system may not IPL properly.

```
ENTER PRINTER NUMBER (NNNN):
■
```

Enter the virtual device number of your printer.

The system default is 000E.

```
ENTER DEVICE TYPE (1403, 3203, 3211, 3262, 3800, 3800-1, 3800-3,
4245, 4248):
■
```

Enter the device type of your printer. The system default is 1403.

Note: **3800** defaults to 3800 Model 1.

```
ENTER PUNCH NUMBER (NNNN):
■
```

If you press **ENTER** without typing anything, you receive the system default, which is 000D. The starter system does not require a punch actually attached to it, but it does create a control block for a punch. If you want to use a real punch at a virtual device number other than 000D, type in the device number and press **ENTER**.

```
ENTER DEVICE TYPE (2540P,3525):
■
```

If you press **ENTER**, you receive the system default, which is 2540P. The starter system does not require a punch actually attached to it, but it does create a control block for a punch. If you want to use a punch other than the default, type in the device type and press **ENTER**.

```
ENTER READER NUMBER (NNNN):
■
```

If you press **ENTER**, you receive the system default, which is 000C. The starter system does not require a reader actually attached to it, but it does create a control block for a reader. If you want to use a real reader at a virtual device number other than 000C, type in the device number and press **ENTER**.

```
ENTER DEVICE TYPE (2540R, 2501, 3505):
■
```

If you press **ENTER**, you receive the system default, which is 2540R. The starter system does not require a reader actually attached to it, but it does create a control block for a reader. If you want to use a reader other than 2540R, type in the device type and press **ENTER**.

```
ENTER NUMBER WHERE FIRST TAPE IS MOUNTED (NNNN):
■
```

Enter the device number of the tape device where the product tape will be mounted. The system default is 0580.

```
ENTER DEVICE TYPE (3420, 3422, 3430, 3480):
devtype ■
```

Enter the device type of your tape drive. The system defaults to a device type of 3420.

```
ENTER THE NUMBER OF A SECOND TAPE DRIVE (NNNN):
■
```

You can type in a device number or press **ENTER**, in which case you receive the default device number, 0581. The starter system does not require a second tape drive actually attached at 0581 or any other device number; the system does create a control block for the device.

```
ENTER DEVICE TYPE (3420, 3422, 3430, 3480):
devtype ■
```

Enter the device type of your second tape drive. The system defaults to a device type of 3420.

```
ENTER DEVICE NUMBER OF WORK PACK (NNNN):
■
```

Enter the device number of your second pack (XASERV). The system default is 0230.

This pack is required for all DASD types **except** double- or triple-density 3380 DASD (3380-E4 and 3380-K) or 3390 DASD.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
■
```

Enter the device type you are using.

```
ENTER NUMBER WHERE EXTRA WORK PACK IS MOUNTED (NNNN):
■
```

Enter the device number of your third pack (XAP001). **The 3350 and 3375 systems require this pack.** The 3380 and 3390 systems do not require an extra work pack.

Note: If you do not have XAP001, you will receive this message:

```
HCPLND108E MAINT 0124 NOT LINKED
```

when you log on as MAINT. If you are using 3380 or 3390 DASD, you can ignore this message.

If you press **ENTER**, you receive the system default, which is 0231.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
■
```

Enter the device type you are using.

3350 DASD Only

```
ENTER NUMBER WHERE EXTRA WORK PACK IS MOUNTED (NNNN):
vdevno ■
```

Enter the device number of your fourth pack (XAP002). **The 3350 system requires this pack.**

If you press **ENTER**, you receive the system default, which is 0232.

```
ENTER DEVICE TYPE (3350, 3375, 3380, 3390):
3350 ■
```

End of 3350 DASD Only

```
ENTER NUMBER OF A DISPLAY DEVICE (NNNN):
vdevno ■
```

This step allows you to define an additional display device. This is **not** the console you are presently using. The system default is 021.

Check your installation's HCPRIO file to see if you have defined this display differently.

```
ENTER DEVICE TYPE (3277, 3278, 3279):
■
```

Enter the device type of your display. The system default is 3278.

```
***SYSTEM DEFINITION COMPLETED***
000E PRINTER
000D PUNCH
000C READER
0580 FIRST TAPE
0581 SECOND TAPE
0230 WORK PACK
0231 EXTRA WORK PACK
0232 OTHER WORK PACK
rdevno GRAPHIC DEVICE
ARE THE ABOVE ENTRIES CORRECT (YES, NO):
```

The numbers shown are those that would appear if you always take the default.

You will see this message only if you are using the 3350 starter system.

yes ■

Answer **yes** to this question. If you respond **no**, the procedure restarts at the first prompt of this step.

5. The starter system continues:

```
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;

SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss, LOADED FROM XASRES
******************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
******************************************

HCPISU951I CP VOLID XASERV NOT MOUNTED
HCPISU951I CP VOLID XAP001 NOT MOUNTED
```

You may receive the *volume* not mounted messages.

```
START ((COLD|WARM|FORCE) (DRAIN) (DISABLE) (NODIRECT))|(SHUTDOWN)
cold drain ■
```

Because there is no data or accounting information to recover, you must request a cold start.

```
NOW hh:mm:ss EST weekday mm/dd/yy
CHANGE TOD CLOCK (YES|NO)
no ■
```

Answer **no**. At second level, you do not have access to the physical hardware needed to change the TOD (Time of day) clock.

```
THE DIRECTORY ON VOLUME XASRES AT ADDRESS nnnn
   HAS BEEN BROUGHT ONLINE.
HCPLND108E MAINT 0124 NOT LINKED;
            VOLID XAP001 NOT MOUNTED
HCPLND108E MAINT 0125 NOT LINKED;
            VOLID XASERV NOT MOUNTED
```

CP logs on the primary system operator (user ID MAINT). These are informational messages for the real system operator. The FILES message refers to spool files; there are no spool files in the system, since this is a cold start of the system.

```
HCPLND108E MAINT 0126 NOT LINKED;
            VOLID XAP002 NOT MOUNTED
THERE IS NO LOGMSG DATA
FILES:      NO RDR,      NO PRT,      NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy
```

You will see this message only if you are using the 3350 starter system.

```
HCPIOP951I CP VOLID volid NOT MOUNTED
⋮
HCPIOP951I USER VOLID volid NOT MOUNTED
```

You may receive a number of informational messages that certain volume identifiers (CP *volids* and user *volids*) are not mounted. You may ignore these messages at this time.

```
STORAGE = 0016M
FILES: 0000001 RDR, 0000001 PRT, NO PUN
```

This message tells you the amount of virtual storage available. You must have 16 megabytes.

6. Initialize CMS and enable all displays:

```
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM

READY; T=n.nn/n.nn hh:mm:ss
enable all ■
READY; T=n.nn/n.nn hh:mm:ss
```

The words VM READ appear in the bottom right corner of the display screen; when VM READ appears, press the **ENTER** key.

This command enables all displays available to your virtual machine. The VM/XA System Product logo appears on all enabled displays.

7. Spool MAINT's console:

```
spool cons * start ■
READY; T=n.nn/n.nn hh:mm:ss
```

This command creates a spool file of all display output for the MAINT virtual machine. Spooling MAINT's console can be valuable if problems arise and you want to review the system generation activity. If you issue this command, you will have a record of this activity. When you close the file (SPOOL CONS CLOSE) or log off, the file is automatically sent to your reader.

8. Set the dump to a tape drive or printer address. It is recommended that you use the second tape drive you defined for sending dumps to. If you use a tape drive, mount a scratch tape with a ring on the drive.

```
set dump vdevno ■
{TAPE | PRINTER} devno DUMP UNIT CP IPL
```

Replace the *vdevno* with the virtual device number of your tape drive or printer to free up some of the spool space.

Warning: If you do not perform this step, you may run out of SPOOL space later on. In the event of a SHUTDOWN and restart at any time during this install procedure, you should perform this substep again.

Stopping and Restarting the Installation

From this point on, when installing VM/XA SP at second level, you can stop the installation after the completion of any step by issuing the following command to **disconnect** your **first-level** user ID:

```
PA1
#cp disconnect ■
```

To restart, log on again to your first-level user ID and issue the following commands:

```
terminal conmode 3270 ■    (local terminals only)
begin ■
```

This puts you back where you were when you stopped.

Step 4. Attach the VM/XA System Product Release 2.1 Product Tape (Volume 1) and Load the First Three Files

In this step, you will:

- Mount Volume 1 of the VM/XA System Product product tape on a tape drive
- Attach the product tape drive to MAINT
- Load the first three files from the product tape.

This step takes approximately 5 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Mount volume 1 of the VM/XA System Product product tape (in write-protect mode) on the tape drive that you specified as the first tape in "Step 3. Load the VM/XA System Product Starter System and Define Devices" on page 117. Follow the operation manual for the machine on which you mount the tape.

2. Attach the tape drive to MAINT at virtual device number 0181:

```
attach vdevno * 181 ■
TAPE vdevno ATTACHED TO MAINT 0181
READY; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the virtual device (*vdevno*) to MAINT's virtual machine at virtual device number 0181.

3. Check MAINT's disk accesses. Be sure MAINT's 191 minidisk is the A disk.

```
query search ■
MNT191   191 A    R/W
MNT190   190 S    R/O
READY; T=n.nn/n.nn hh:mm:ss
```

The QUERY SEARCH command tells you what disks you have accessed and the alphabetic order of those accesses. When searching for files, CMS uses the alphabetic order of accesses to create a search order. CMS searches the A minidisk, if accessed, then the B minidisk, if accessed, and so forth.

4. If MAINT's 191 minidisk is not A, access that minidisk as A:

```
access 191 a ■
191 REPLACES A (195)
READY; T=n.nn/n.nn hh:mm:ss
```

This message would appear if the QUERY SEARCH command had shown MNT195 195 A R/W.

5. For your own convenience, set **PF12** as a retrieve key. When you press the retrieve key, the last instruction you issued will be displayed. You need only press **ENTER** to enter it again.

```
set pf12 retrieve ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You may add this command to your PROFILE EXEC.

To Speed Up the Installation Procedure

Creating a PROFILE EXEC and a PROFILE XEDIT file at this time can save you time in subsequent installation steps. You can set up PF key assignments in your PROFILE EXEC to save keystrokes. You can create a PROFILE XEDIT to present the XEDIT screen to your liking instead of taking the default XEDIT assignments provided by IBM.

6. Use the VMFPLC2 LOAD command to load the first three files (the tape header, memos, and installation tools) from the product tape to your A-disk:

```
vmfplc2 load (eof 3 ■
 LOADING.....
 $TAPE$   HEADER   A1
 END-OF-FILE OR END-OF-TAPE
```

This command loads the first three tape files from the product tape. (A tape file is the set of files between two tape marks.) These are the only tape files you have to load—the ITASK EXEC automatically loads the rest of the tape. You will receive a message from VMFPLC2 telling you which files it loaded.

```
 SUPSP21  MEMO     A1
 56643089 SERVICE  A1
 END-OF-FILE OR END-OF-TAPE
```

The VMSUP memo is loaded to the MAINT 191 minidisk. Read this memo before going on.

After you perform Step 8, you may wish to copy SUPSP21 MEMO to the CP BASE disk (194 or 394) to free space on MAINT 191.

```
 56643089 $PPF     A1
 RPWLIST  DATA     A1
 $MSG4I$  EXCAMENG A1
 $$RDPW$$ EXEC     A1
 $MSG4I$  EXEC     A1
 SETUP    EXEC     A1
 SPLOAD   EXEC     A1
 UTILITY  EXEC     A1
 ITASK    EXEC     A2
 DIRECTXA MODULE   A2
 DVM      PROFILE  A1
 SPLOAD   PROFILE  A1
 END-OF-FILE OR END-OF-TAPE
```

The tools in the third file are installation EXECs. You may erase them **after you finish the installation**, except for these files:

- DIRECTXA MODULE, which is erased during the installation procedure
- SETUP EXEC, which **must** remain on the MAINT 191 minidisk.

The DIRECTXA MODULE is loaded to MAINT 191 now so that you can bring your directory online in Step 5. It is erased after it is used, then reloaded to MAINT 190 in Step 8. If you need to reissue ITASK ALLOCATE, you must rewind the tape and issue VMFPLC2 LOAD DIRECTXA MODULE A (EOT to reload the DIRECTXA MODULE from Volume 1 of the product tape.

Step 5. Invoke the ITASK EXEC

In this step, you will use the ITASK EXEC to:

- Provide a password for user IDs MAINT and OPERATOR
- Load the system samples and the sample user directory
- Format the XASERV, XAP001, and XAP002 volumes.

The time for this step depends upon the devices involved. It usually takes at least 30 minutes for each pack you format.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

Note: A description of the ITASK EXEC and its operands can be found in "ITASK EXEC" on page 611.

Warning: To do the load, ITASK EXEC issues the COPYFILE command with the REPLACE option. This command overwrites any files already on the disk that have the same labels as the sample files being copied to it. If you are redoing this step, and you have already tailored some of the sample files, you may want to rename the files that you tailored before you invoke ITASK.

Warning: The ITASK EXEC cannot handle virtual DASD that are defined with different physical characteristics from the real DASD. Consider this restriction when you define the virtual DASD used in the installation procedure. For example, do not define a real double-density 3380 as two virtual single-density 3380s.

Note: Do this step while logged on as the MAINT user ID.

1. Make your DASD volumes available for use; issue the following command for each volume except XASRES:

   ```
   vary online vdevno ■
   ```

   vdevno is the **first-level virtual** address of the volume; this address functions as a "real" address at second level.

2. Issue ITASK ALLOCATE to supply a password for MAINT and OPERATOR and to format the XASERV, XAP001, and XAP002 volumes:

   Note: The example in this manual uses a 3380 DASD for the minidisks. If you use a 3350, 3375, 3380-E4, 3380-K, or 3390, your space allocations and starting address for CMS will differ.

   ```
   itask allocate ■
   ```

   ```
   REWIND COMPLETE
   DMSACC724I 191 REPLACES A (191)
   REWIND COMPLETE
   DMSWTK409I LOADING 33xx * TO 191
              ATTACHED TO MAINT
   ```

   These are informational messages to tell you how the operation is progressing. *xx* is 50, 75, 80, 8E, 83, 90, or 91, according to your DASD type.

   Note: If ITASK exits with return code 1040, the 123 full-pack minidisk is not attached. The virtual address of XASRES is 123. You may have to reattach XASRES to your system.

3. When prompted, enter a new password to be used as both your MAINT and OPERATOR passwords:

```
ENTER ONE PASSWORD FOR MAINT AND OPERATOR:
password ■
```

You must supply a single password to be used for both the MAINT and OPERATOR user IDs. The ITASK EXEC will XEDIT the user directory and replace NOLOG with the logon password that you provide here.

When selecting a password, refer to the RPWLIST DATA file shown in Appendix F, "Restricted Logon Passwords" on page 859. This file contains all of the restricted passwords for your VM/XA SP2.1 system. You cannot choose a password for MAINT and OPERATOR that exists in this file.

```
ENTER PASSWORD AGAIN TO VERIFY:
password ■
```

Reenter the password that you have designated for the MAINT and OPERATOR user IDs.

YOU NEED TO REMEMBER THIS PASSWORD FOR USE IN LATER STEPS. You can write it down on the worksheet—see Table 1 on page 7.

```
VM/XA SYSTEM PRODUCT USER DIRECTORY
  CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ON LINE
DMSWSL970I FORMATTING MAINT 193 MINIDISK
DMSWSL409I LOADING SYSGEN TOOLS TO
      THE 193 DISK ATTACHED TO MAINT
```

These first messages signify that the user directory has been created. Then ITASK loads the system generation tools needed next.

```
DMSWSL970I FORMATTING MAINT 295 MINIDISK
DMSWSL409I LOADING HCPSYSxx * TO
           THE 295 DISK ATTACHED TO MAINT
DMSWSL409I LOADING HCPRIOXA * TO
           THE 295 DISK ATTACHED TO MAINT
DMSWSL409I LOADING HCPBOX * TO
           THE 295 DISK ATTACHED TO MAINT
DMSWSL970I FORMATTING MAINT 395 MINIDISK
DMSWSL409I LOADING CMSLOC SAMPLES TO
           THE 395 DISK ATTACHED TO MAINT
DMSACC724I 191 REPLACES A (191)
```

These are messages from ITASK as it loads files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

```
DMSWTK965I YOU MAY WISH TO TAILOR THE FOLLOWING
           FILES BEFORE NUCLEI GENERATION:

           HCPRIO ASSEMBLE
           HCPBOX ASSEMBLE
           HCPSYS ASSEMBLE
           DMSNGP ASSEMBLE
           USER  DIRECT
```

You will be able to tailor HCPRIO ASSEMBLE, HCPBOX ASSEMBLE, and HCPSYS ASSEMBLE in "Step 17. Tailor the Sample HCPRIO, HCPSYS, HCPBOX Files and the Product Parameter File" on page 158. You will tailor DMSNGP ASSEMBLE in "Step 14. Tailor the DMSNGP Profile" on page 148. You will tailor USER DIRECT in "Step 21. Update the User Directory" on page 172.

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XASERV VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

vdevno or skip ■

Enter the virtual address, *vdevno*, of the XASERV volume. Remember that the virtual address functions as a real address at second level. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you do not want to use XASERV, or if you have already formatted it, enter **skip**.

```
DASD vdevno ATTACHED TO MAINT 301
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 301
```

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 301
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
CYLINDERS nnnn THROUGH nnnn FORMATTED
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XASERV'
```

This message signifies that CPFMTXA has completed formatting the XASERV volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED MAINT 0301
DASD vdevno ATTACHED TO SYSTEM XASERV
```

The HCPFAM384I message signals the completion of this sequence.

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP001 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

vdevno or skip ■

```
DASD vdevno ATTACHED TO MAINT 302
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 302
```

Enter the **virtual** address, *vdevno*, to allocate the XAP001 volume. **Check that you typed the right address before you press ENTER**.

The 3350 and 3375 starter systems require XAP001. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume.

If you do not want to use XAP001, or if you have already formatted it, enter **skip**.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 302
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP001'
```

This message signifies that CPFMTXA has completed formatting the XAP001 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED MAINT 0302
DASD vdevno ATTACHED TO SYSTEM XAP001
```

The HCPFAM384I message signals the completion of this sequence.

3350 DASD Only

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP002 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

```
vdevno or skip ■
DASD vdevno ATTACHED TO MAINT 303
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 303
```

Enter the virtual address, *vdevno*, to allocate the XAP002 volume. The 3350 starter system requires XAP002. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you have already formatted XAP002, enter **skip**.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 303
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP002'
```

This message signifies that CPFMTXA has completed formatting the XAP002 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED MAINT 0303
DASD vdevno ATTACHED TO SYSTEM XAP002
```

The HCPFAM384I message signals the completion of this sequence.

End of 3350 DASD Only

```
READY; T=n.nn/n.nn hh:mm:ss
```

This message tells you that the ITASK EXEC has completed the sequence for each volume needed for installing VM/XA System Product Release 2.1.

Step 6. Format the Remaining Base CP Minidisks

In this step, you will use the ITASK EXEC to format the remaining minidisks that are defined in the CP directory, but have not yet been formatted or loaded in a previous step. ITASK also puts a PROFILE EXEC on the 191 minidisk for EREP and DISKACNT.

The time to perform this step may vary, but a good estimate is 45 minutes.

1. Use the ITASK EXEC with the BASEIDS parameter to format, load, and add (as appropriate) PROFILE EXECs to the remaining minidisks in the base CP directory. If you issue ITASK BASEIDS from a user ID other than MAINT, it will format that user ID's minidisks instead of MAINT's.

   Note: The block sizes of the 190 and 490 minidisks have increased from 1KB to 4KB for minidisk caching. The Starter System defines the 190 and 191 minidisks (formatting the 191 disk with a 1KB block size), while the ITASK BASEIDS step formats all other minidisks with a 2KB block size.

   If you choose to format a minidisk with a block size of 4KB, the number of cylinders for that minidisk may have to be increased, as is the case for the HELP disks.

```
itask baseids ■
DMSWTK1310I DO YOU WANT TO FORMAT MINIDISKS:
             AUTOLOG1 (191), CMSBATCH (195), DISKACNT (191),
             MAINT 36E (PVM), EREP (191), AND OPERATNS (191)?
DMSWTK8005R TYPE: 1 (YES) OR 0 (NO).
1 ■

DMSWTK968I THE FOLLOWING MINIDISKS
           DEFINED IN THE BASE CP DIRECTORY
           WILL BE FORMATTED:

           MAINT    292     MAINT    49C
           MAINT    194     MAINT    49D
           MAINT    19C     MAINT    501
           MAINT    19D     MAINT    591
           MAINT    19E     MAINT    592
           MAINT    201     MAINT    593
           MAINT    291     MAINT    594
           MAINT    293     MAINT    595
           MAINT    294     MAINT    596
           MAINT    391     MAINT    59E
           MAINT    392     MAINT    5E5
           MAINT    393     MAINT    691
           MAINT    394     MAINT    692
           MAINT    423     MAINT    791
           MAINT    490     MAINT    892
           MAINT    491     MAINT    895
           MAINT    492     MAINT    896
           MAINT    495     MAINT    89E

DMSWTK970I FORMATTING AUTOLOG1 191 MINIDISK
DMSWTK970I FORMATTING CMSBATCH 195 MINIDISK
DMSWTK970I FORMATTING DISKACNT 191 MINIDISK
DMSWTK969I A PROFILE EXEC HAS SUCCESSFULLY
```

```
      BEEN COPIED TO DISKACNT'S 191 MINIDISK
DMSWTK970I FORMATTING EREP 191 MINIDISK
DMSWTK969I A PROFILE EXEC HAS SUCCESSFULLY
      BEEN COPIED TO EREP'S 191 MINIDISK
DMSWTK970I FORMATTING MAINT 36E MINIDISK
DMSWTK970I FORMATTING OPERATNS 191 MINIDISK
DMSWTK970I FORMATTING MAINT 292 MINIDISK
DMSWTK970I FORMATTING MAINT 194 MINIDISK
DMSWTK970I FORMATTING MAINT 19C MINIDISK
DMSWTK970I FORMATTING MAINT 19D MINIDISK
DMSWTK970I FORMATTING MAINT 19E MINIDISK
DMSWTK970I FORMATTING MAINT 201 MINIDISK
DMSWTK970I FORMATTING MAINT 291 MINIDISK
DMSWTK970I FORMATTING MAINT 293 MINIDISK
DMSWTK970I FORMATTING MAINT 294 MINIDISK
DMSWTK970I FORMATTING MAINT 391 MINIDISK
DMSWTK970I FORMATTING MAINT 392 MINIDISK
DMSWTK970I FORMATTING MAINT 393 MINIDISK
DMSWTK970I FORMATTING MAINT 394 MINIDISK
DMSWTK970I FORMATTING MAINT 423 MINIDISK
DMSWTK970I FORMATTING MAINT 490 MINIDISK
DMSWTK970I FORMATTING MAINT 491 MINIDISK
DMSWTK970I FORMATTING MAINT 492 MINIDISK
DMSWTK970I FORMATTING MAINT 495 MINIDISK
DMSWTK970I FORMATTING MAINT 49C MINIDISK
DMSWTK970I FORMATTING MAINT 49D MINIDISK
DMSWTK970I FORMATTING MAINT 501 MINIDISK
DMSWTK970I FORMATTING MAINT 591 MINIDISK
DMSWTK970I FORMATTING MAINT 592 MINIDISK
DMSWTK970I FORMATTING MAINT 593 MINIDISK
DMSWTK970I FORMATTING MAINT 594 MINIDISK
DMSWTK970I FORMATTING MAINT 595 MINIDISK
DMSWTK970I FORMATTING MAINT 596 MINIDISK
DMSWTK970I FORMATTING MAINT 59E MINIDISK
DMSWTK970I FORMATTING MAINT 5E5 MINIDISK
DMSWTK970I FORMATTING MAINT 691 MINIDISK
DMSWTK970I FORMATTING MAINT 692 MINIDISK
DMSWTK970I FORMATTING MAINT 791 MINIDISK
DMSWTK970I FORMATTING MAINT 892 MINIDISK
DMSWTK970I FORMATTING MAINT 895 MINIDISK
DMSWTK970I FORMATTING MAINT 896 MINIDISK
DMSWTK970I FORMATTING MAINT 89E MINIDISK

DASD 333 DETACHED

READY; T=n.nn/n.nn hh:mm:ss
```

The ready message indicates that ITASK has completed its work successfully.

2. If any of the minidisks listed in message DMSWTK968I were not formatted, check your work packs to make sure that they are labeled XASERV, XAP001, and XAP002, and that they are attached to the system. Then reissue ITASK BASEIDS.
3. Continue with the next step.

Step 7. ITASK Loads Files from the Product Tape (Volume 1)

In this step, you will issue the ITASK EXEC to load files from Volume 1 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Enter the ITASK command with the LOAD ALL1 operands to load the CP and dump viewing facility object code and service files. If a problem should occur during this step, or for if some reason you want to return to this step and load product tape files, see "ITASK EXEC" on page 611 for instructions on loading the files you need.

```
itask load all1 ■
DMSWSL409I LOADING SYSTEM SAMPLES TO
     THE 191 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP OBJECT TO
     THE 194 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP PUTAPPLY TO
     THE 292 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP PUTDELTA TO
     THE 294 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP CORAPPLY TO
     THE 292 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CP CORDELTA TO
     THE 294 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW OBJECT TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW PUTAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW PUTDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW CORAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING DUMPVIEW CORDELTA TO
     THE 293 DISK ATTACHED TO MAINT
READY; T=n.nn/n.nn hh:mm:ss
```

These are messages from ITASK as it loads more files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

Step 8. ITASK Loads Files from the Product Tape (Volume 2)

In this step, you will issue the ITASK EXEC to load files from Volume 2 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match.** If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Mount Volume 2 of the product tape.
2. Issue ITASK LOAD CMS to load the CMS object code, source code, and service files and the IOCP files:

```
itask load cms ■
DMSWSL409I LOADING CMS BASE TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING IOCP FILES TO
     THE 193 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS PUTAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS PUTDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS CORAPPLY TO
     THE 392 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS CORDELTA TO
     THE 293 DISK ATTACHED TO MAINT
DMSWSL409I LOADING CMS SYSTEM TO
     THE 190 DISK ATTACHED TO MAINT
```

3. The ITASK EXEC invokes the SETUP EXEC to establish the minidisk access order, then the ASMGEND EXEC to build the system F assembler and create the associated auxiliary directory:

```
DMSACC724I 191 REPLACES A (191)
⋮
DMSACC724I 190 REPLACES A (191)
DMSACC725I 190 ALSO = S DISK
ASSEMBLE XF GEND PROC
ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED.
■
```

Press **ENTER** to place the modules on the S-disk, the system default.

Enter another disk-mode letter if you prefer the modules to be placed on another minidisk.

```
ASSEMBLE XF GEND COMPLETE

DMSACC724I 191 REPLACES A (190)
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

a. At this time, you may choose to make a copy of the documentation file (SUPSP21 MEMO) to the CP BASE disk as suggested in Step 4, substep 6 on page 125.

b. You may choose to load HELP files, CP, CMS, and dump viewing facility source files, and GCS object files at this time. The following is an alternative order for loading these files that will minimize tape mounts:

   1) Load HELP files from Volume 2. (See Step 22 on page 173.)

   2) Load GCS object files from Volume 2. (See Step 26, substep 3 on page 179.)

   3) Load CP, CMS, and dump viewing facility source files from Volume 3. (See Step 23 on page 174.)

Step 9. Establish the Service Tools Build Disk

In this step, you will copy the latest level of the service tools to the service tools build disk.

This step takes approximately 10 to 15 minutes.

To be sure that you are using the most recent service EXECs, you must have the latest level of service on a service tools build disk. This disk, 5E5, is defined in the sample directories. It is the only disk in the TASK string in the product parameter file.

1. IPL 190:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order and create a work EXEC to help you copy the service tools:

```
setup ■
Ready; T=n.nn/n.nn hh:mm:ss
DMSACC724I vdevno replaces fm (vdevno
⋮
vmfsetup 56643089 cms ■
DMSACC724I vdevno replaces fm (vdevno
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
xedit lvlses exec fm ■
input ■
```

fm is the filemode of the task disk (5E5).

```
&TRACE OFF
&FT = &LEFT OF &2 2
&FT = &CONCAT OF &FT *
EXEC FILELIST &1 &FT *
&EXIT 0
```

Enter these lines exactly as they appear.

```
■
■
file ■
```

Press **ENTER** twice to return to the command line.

3. Look for the latest version of the CPYSVSES EXEC or CPYSVSES EXC*nnnnn*. CPYSVSES contains a list of all the service EXECs.

```
filelist cpysvses * * ■
```

Examine each version of CPYSVSES. CPYSVSES has an update history at the top. The latest level is one with the most updates. Use the update history, **not the date and time stamp**, to determine which is the most recent version.

4. Copy the latest level of CPYSVSES to the task disk (5E5) with a filetype of EXEC:

```
copyfile / = exec fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

5. Invoke CPYSVSES EXEC:

```
cpysvses exec lvlses ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. CPYSVSES EXEC invokes LVLSES, which invokes FILELIST to display each entry for each service EXEC listed in CPYSVSES. Process each FILELIST screen as follows:

   a. For each of the files on the FILELIST screen, determine which is the latest version. **Do not depend on the date and time stamps;** examine the actual files. Each file ends with an update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn*, MOD*nnnnn*, XED*nnnnn*, and PRF*nnnnn* are exact images of the corresponding EXEC, MODULE, XEDIT, and PROFILE executables.

   b. Copy the latest files to the task disk (5E5), using the OLDDATE REPLACE options. Assign the copy a filetype of EXEC, MODULE, XEDIT, or PROFILE:

```
copyfile / = ft fm (olddate replace ■
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

   c. Press **PF3** to exit this FILELIST screen and go to the next screen.

   d. Continue until there are no more FILELIST screens.

7. Look for the latest version of the product parameter file:

```
filelist 56643089 * * ■
```

Examine each version of the product parameter file. A sample product parameter file is shown in "A Sample Product Parameter File" on page 384. The product parameter file has an update history at the top. The latest level is the one with the most updates. Use the update history, **not the date and time stamp**, to determine which is the most recent version.

8. Copy the latest level of the product parameter file to the task disk (5E5):

```
copyfile / = $ppf fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

9. Now that the service tools are all on the service tools build disk, you do not need to issue SETUP again. Instead, simply access 5E5 as B. You may wish to add this access to MAINT's PROFILE EXEC.

```
access 5E5 b ■
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase any duplicate files from the MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

`filelist * * b` ■	This command lists all the files on your B-disk.
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.
`erase` ■	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the B-disk (5E5) or any other disk on which it may appear.

Step 10. Set the System Default National Language

In this step, you will set the system default national language.

This step takes approximately 20 minutes.

The system default national language is mixed-case American English. This means that most system messages and HELP files will appear in mixed case (capital and small letters). Some will appear in uppercase (capital letters only).

Decide whether you want to keep the system default as mixed-case American English or change it to uppercase American English (in which case all system messages and HELP files will be in uppercase). Follow the appropriate instructions below.

If you want to install a system default national language other than American English, you will be able to do so after you finish the installation process. See Chapter 5, "Installing a New System National Language" on page 311 for instructions.

If you want to install an alternate system national language (including whichever version of American English is not your default), see *VM/XA SP Planning and Administration.*

Mixed-Case American English

If you want to keep the system default national language as mixed-case American English, you should convert the installation messages to mixed case. (If you do not do so, all the installation messages will continue to appear in uppercase, but most other system messages will appear in mixed case.)

1. Save the uppercase installation message file:

```
rename $msg4i$ exec a = excuceng = ■
READY; T=n.nn/n.nn hh:mm:ss
```

2. Rename the mixed-case installation message file:

```
rename $msg4i$ excameng a = exec = ■
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSAMENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $MSG4I$ EXEC, the service will be in uppercase. You do not have to convert it to mixed case.

Uppercase American English

If you want to change the system default to uppercase American English, you must:

1. Edit the product parameter file and change all :NLS. tags from AMENG to UCENG:

```
xedit 56643089 $ppf ■
ch/:NLS.      AMENG/:NLS.      UCENG/* * ■
file ■
```

There are six blanks after :NLS. in the XEDIT change command.

2. Save DMSNGP TEXT, then rename the DMSNGP TXTUCENG file DMSNGP TEXT. This gives you a TEXT file assembled from a DMSNGP profile with LANGID=UCENG and HELP=19C.

   Note: You will be able to make other changes in the DMSNGP profile in Step 14, after you install Assembler H. You will have to change the LANGID= and HELP= parameters in Step 14. The DMSNGP TXTUCENG file is provided here only to allow you to specify uppercase American English as your system default national language.

```
access 395 d ■
DMSACC724I 395 REPLACES D(395)
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp text d = txtameng = ■
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp txtuceng d = text = ■
READY; T=n.nn/n.nn hh:mm:ss
```

3. Convert $VMFMSG$ EXEC (the service message file) to uppercase:

```
xedit $vmfmsg$ exec ■
uppercas * ■
file ■
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSUCENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $VMFMSG$ EXEC, the service will be in mixed case. You will have to convert it to uppercase. If you receive service for the DMSNGP profile or the product parameter file, you will have to update them again.

Step 11. Build CMS

In this step, you will build CMS.

This step takes approximately 5 minutes.

1. IPL 190 and issue ITASK BUILD CMS to invoke the CMS BUILD process:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
itask build cms ■
nnnnnnnn files changed
DMSACC724I 191 replaces A(191)
DMSACC724I 5E5 replaces B(5E5)
⋮
```

ITASK invokes the SETUP EXEC to create the correct CMS access order.

```
DMSWSU1900W The existing 56643089 $SETUP A1
            file has been refreshed.  You
            might want to check your access
            order when done.
⋮
```

You will see this message only if you have a copy of 56643089 $SETUP on your A-disk. If you do see it, you will get return code 4.

```
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

A temporary load list is created and the system loader is invoked.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
  RECS nnnK COPY 001 NOHOLD NOKEEP
```

The CMS nucleus has been sent to your reader.

```
DMSWSU1906E The access of 19E failed with a
            return code of 28.  Processing
            stops for product 56643089.
            The original access order cannot
            be restored.
```

You may receive this message. It is not a problem. The access failed because 19E is empty. You will reaccess 19E when you re-IPL 190. If you see this message, you will get return code 4.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

2. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

3. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 7 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

4. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

5. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED   KEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

6. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

7. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

```
HCPLDR8013I Possible overlay:  .SLC 000000
```

If you see either of these messages, you can ignore them. They are informational messages.

```
DMSINS283E The NLSlangid saved segment could not be found;
            return code 44 from SEGMENT
```

```
DMSINQ609R Nucleus (CYL or BLK) address =
cyl ■
```

Enter the correct starting cylinder address for the DASD type of your 190 minidisk:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

```
DMSINQ612R Enter version identification:
■
```

Press **ENTER** for the default version identification, VM/XA CMS 5.6 *mm/dd/yy hh:mm*. You will be able to modify the default in Step 14.

```
DMSINQ612R Enter installation heading:
■
```

Press **ENTER** for the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM. You will be able to modify the default in Step 14.

```
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

Informational message: The optional installation segment (CMSINST) is not loaded and saved until "Step 29. Create the CMSINST and HELP Saved Segments" on page 204 is completed.

Step 12. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to MAINT's virtual printer. Ready the printer to send the CMS load map to MAINT's virtual reader:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt * ■

RDR FILE fileno SENT FROM MAINT
PRT AS fileno RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The CMS load map has been sent to MAINT's virtual reader. If you do not get this message, it has already been sent there by a previous IPL 190.

2. If the CMS load map is sent to MAINT's virtual printer instead of the reader, transfer it to the virtual reader:

```
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Identify the reader file:

```
query rdr * all ■

ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,100 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

4. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft d ■
File fn ft D received from MAINT at * sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

5. Print a copy of the CMS load map. If you do not have a printer attached, skip this subset.

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Examine the load map for errors:

```
xedit fn ft ■
:
qquit ■
```

7. Continue with the next step.

Step 13. Install Assembler H Version 2 Program Product

Note: IBM packages Assembler H Version 2 as a separate program product. This program product has its own installation procedure.

In this step, you will:

- Consult *Assembler H Version 2 Program Product: Installation*, SC26-4030
- Install Assembler H Version 2 Program Product, VM feature
- Install service for Assembler H Version 2.

The time to perform this step may vary, but a good estimate is 20 minutes.

1. Before you install Assembler H, consult your IBM Support Center for the latest information on APARs.
2. Read the *Installation* manual for Assembler H Version 2 Program Product.
3. Refer to "Installing Program Products" in the *VM/XA SP2.1 Program Directory* for more information on Assembler H.
4. Follow the directions in the *Installation* manual to install Assembler H Version 2. IBM advises you to install Assembler H Version 2 on the 19E minidisk.

   Note: Use the OBJECT option to assemble Assembler H CSECTS.
5. During the installation of Assembler H Version 2, answer YES to the question "BUILD AN AUXILIARY DIRECTORY" and respond Y to the question asking what mode letter you want to use to access the disk containing the H-Assembler modules.
6. Erase IEV61 TEXT. If you do not erase this file, you will not be able to apply service to Assembler H.
7. Install the latest service level of Assembler H.
8. After you have finished installing service for Assembler H, IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase any duplicate files from the MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

`filelist * * fm ■`	This command lists all the files on the task disk (5E5).
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.

	erase ■	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the task disk (5E5).

10. If any unneeded service memos exist on the MAINT 191 disk, erase them to make room for further processing.

Step 14. Tailor the DMSNGP Profile

In this step, you will use the VM System Product Editor (XEDIT) supplied with the starter system to tailor the CMS nucleus generation profile (DMSNGP).

This step, excluding time for printing system definition files, takes approximately 15 minutes.

1. Before proceeding with this step, read the discussion of the DMSNGP profile in *VM/XA SP Planning and Administration*, under the discussion of the DEFNUC macro. DMSNGP ASSEMBLE is a profile that contains CMS configuration defaults and responses to system prompts; these will become part of the CMS nucleus.

2. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Access the CMS LOCAL1 minidisk, where DMSNGP resides, as D:

```
access 395 d
:
Ready; T=n.nn/n.nn hh:mm:ss
```

In this example the DMSNGP file resides on the 395 disk.

4. Save a copy of DMSNGP ASSEMBLE, so that if you make a mistake in changing it, you can go back to the original file:

```
copyfile dmsngp assemble d = oldassem = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. XEDIT the CMS nucleus generation profile:

```
xedit dmsngp assemble ■
```

Within the file, you can use the normal XEDIT cursor-control keys, scroll keys, insert key, and delete key.

```
NGP      TITLE 'DMSNGP     (CMS)      VM/XA SYSTEM PRODUCT 5664-308'
         EJECT
DMSNGP   CSECT
         DEFNUC  SYSDISK=190,          * S disk address                *
                 YDISK=19E,            * Y-disk address                *
                 HELP=19D,             * Help disk address             *
                 LANGID=AMENG,         * Default is American English   *
                 DBCS=NO,              * Default is not a DBCS lang    *
                 LANGLEV=S,            * DCSS ID for multiple DCSS     *
                 SAVESYS=NO,           * Using CMS in DCSS YES OR NO   *
                 SYSNAME=CMS,          * Name of above DCSS to save    *
                 USEINST=YES,          * Using EXEC/XEDIT in DCSS      *
                 INSTSEG=CMSINST,      * Name of above DCSS to save    *
                 REWRITE=YES,          * Write nucleus yes or no       *
                 IPLADDR=190,          * Address of where to write     *
                 CYLADDR=?,            * CYL/BLK OF WHERE TO WRITE     *
                 IPLCYL0=YES,          * Write IPL text on cyl 0       *
                 VERSION=?,            * VM/XA CMS 5.6 MM/DD/YY HH MM SS*
                 INSTID='VM/XA CONVERSATIONAL MONITOR SYSTEM'
         END
```

6. Look at the DEFNUC defaults, and make any necessary changes. Refer to *VM/XA SP Planning and Administration* for information on the DEFNUC macro.

   Warning: The comma at the end of each line of the DEFNUC macro is required in all lines but the last line. If you omit this comma in any line other than the last line, all the lines after the first line without a comma are ignored.

   Warning: On each line, make sure that you leave at least one blank between the comma and the comment.

   - At a minimum, you should change **CYLADDR=?**, to indicate the starting cylinder on the MAINT 190 minidisk at which to start writing the CMS nucleus. The appropriate starting cylinder depends on the device type of the DASD where the MAINT 190 minidisk is defined:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

     If you do not replace the question mark for the CYLADDR keyword, you receive the following prompt when you generate the CMS nucleus:

     ```
     DMSINI609R Nucleus (CYL or BLK) address =
     ```

     You must respond with the appropriate cylinder address.

   - If you specify **YES** after the SAVESYS option, you must create a skeleton named saved segment (NSS) before building the CMS nucleus. This segment is saved when you build CMS and IPL your reader. (You will get message HCPNSS440I.)

     A skeleton segment for the CMS nucleus is created by the SAMPNSS EXEC. You can also create your own skeleton segment with the DEFSYS command. For information on the DEFSYS command, see *VM/XA SP CP Command Reference* and *VM/XA SP Virtual Machine Operation.*

- Make sure the LANGID and HELP parameters are right for your system default national language:

Language	LANGID	HELP
Mixed-case American English	AMENG	19D
Uppercase American English	UCENG	19C

- Make any other changes you wish. For example, you may want to change the version identifier or the installation header. You must put **single quotes** around your version identifier and installation header. The character string between the single quotes can be any combination of alphanumeric characters.

```
file = = fm ■
```

Make your changes and file the update file on the CMS LOCAL1 (395) disk.

7. Assemble the new DMSNGP file:

```
itask assemble dmsngp ■
:
DMSUPD181E No update files were found
DMSWHM1907I Assembling DMSNGP
```

The minidisk access order is reestablished.

Note: If an error occurs during the assembly, examine the flagged statements and correct the statements in error. Then continue from the beginning of this step.

```
DMSWHM1909I DMSNGP TEXT A created
PRT FILE fileno SENT FROM MAINT   PUN AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

The DMSNGP file has been assembled and placed on the MAINT 191 minidisk.

8. Copy the text deck from MAINT 191 to MAINT 395, then erase it from MAINT 191:

```
copy dmsngp text a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase dmsngp text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the filemode of the CMS LOCAL1 minidisk.

If you did not replace the question mark for the CYLADDR keyword, when you generate the CMS nucleus, you will receive the following message:

Message	Appropriate Response
DMSINI609R Nucleus (CYL or BLK) address =	*nnnnn* *nnnnn* is the location on the 190 disk where the new CMS nucleus is written. Refer to the table on page 149 to determine the correct response for your DASD type.

Step 15. ITASK EXEC Rebuilds the CMS Nucleus

Note: Do this step only if you changed the DMSNGP file in Step 14.

In this step, the ITASK EXEC:

- Creates a CMS nucleus
- Places the CMS nucleus on the CMS residence disk.

This step takes approximately 5 minutes.

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Issue ITASK BUILD CMS to create the CMS nucleus. By issuing the ITASK EXEC, you will incorporate updated CMS text files into a new CMS nucleus.

```
itask build cms ■
nnnnnnnn files changed
⋮
```

ITASK invokes the SETUP EXEC to create the correct CMS access order.

```
DMSWSU1900W The existing 56643089 $SETUP A1
              file has been refreshed.  You
              might want to check your access
              order when done.
```

You will see this message and return code 4 only if you did not erase 56643089 $SETUP in Step 11.

```
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

A temporary load list is created, and the system loader is invoked.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
  RECS nnnK COPY 001 NOHOLD NOKEEP
```

The CMS nucleus has been sent to your reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file you will IPL in substep 8 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
HCPLDR8015I POSSIBLE OVERLAY:  .SLC 000000
```

You may receive this message. It can be ignored.

CMS Nucleus Generation Prompts and Responses

Note: Each of the following prompts appears **only** if the corresponding statement in DMSNGP is missing or empty (and the DEFNUC macro contains no default value), or if the DMSNGP statement contains a question mark (?):

```
DMSINQ606R System disk address =
■
```

190 is the default.

```
DMSINQ615R Y-disk address =
■
```

19E is the default.

```
DMSINQ640R HELP disk address =
■
```

19D is the default.

```
DMSINQ764R Language id =
■ or langid ■
```

This response identifies the *langid* of your system national language. The default *langid* is AMENG (mixed-case American English).

```
DMSINQ293R Is this a DBCS language?  Enter 1 (YES) or 0 (NO).
```

```
0 ■
```

The default is 0 (NO); mixed-case American English is not a DBCS (Double-Byte Character Set) language.

```
DMSINQ295R Language level id =
```

```
■
```

The system national language does not use a level ID.

```
DMSINQ296R Should the installation segment be used?  Enter 1 (YES) or 0 (NO).
```

The installation segment is an optional shared saved segment, into which you can place frequently used EXECs and System Product Editor (XEDIT) macros. You install the segment **after** you install your base system, but you must indicate now whether or not you are going to use it.

```
■ (or 1 ■) or 0 ■
```

The default is 1 (YES). Enter **0** if you do not want to use the segment.

```
DMSINQ310R Installation segment name =
```

This prompt appears only if you accepted the default (or entered **1**) at the previous prompt.

```
■ or segname ■
```

Enter a 1- to 8-alphanumeric-character name for the installation segment, or press **ENTER** to accept the default name, CMSINST.

```
DMSINI729R Do you want to save the system?  Enter 1 (YES) or 0 (NO).
```

```
0 ■
```

Answer **0** (NO). You will save CMS in Step 27.

The default is 1 (YES).

Warning: The default response to this prompt does not agree with the sample DMSNGP file, where SAVESYS=NO.

```
DMSINI730R Saved system name =
```

You will not see this prompt if you answered **0** to the preceding prompt.

```
■ or sysname ■
```

The default system names are CMS and CMSXA.

```
DMSINI607R Rewrite the nucleus?  Enter 1 (YES) or 0 (NO).
```

```
1 ■
```

Enter **1** to write the CMS nucleus on the disk that you specify in your response to the next prompt.

```
DMSINI608R IPL device address =
```

```
■
```

The default is the address of the system disk (190).

```
DMSINI609R Nucleus (CYL or BLK) address =
```

nnn ■

nnn is the location on the 190 system disk where the new CMS nucleus is written. Enter the correct cylinder/block address for your DASD type:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

```
DMSINI610R Also IPL CYL/BLK 0?  Enter 1 (YES) or 0 (NO).
```

1 ■

Enter **1** to write the initial IPL text on cylinder/block 0 of the disk where the CMS nucleus is written.

The initial IPL text is a bootstrap program that reads the CMS nucleus from the cylinder/block where the nucleus is written (as defined in your response to prompt DMSINI609R). The initial IPL text is always written on the same cylinder/block as the nucleus. If the initial IPL text is not also written on cylinder/block 0, you must specify the cylinder/block address of the nucleus when you issue IPL commands for this system. For more information, refer to the description of the IPL command in *VM/XA SP CP Command Reference*.

```
DMSINI611R Enter version identification:
```

The version identification is displayed each time that you IPL the CMS system you are now generating.

■ or *version* ■

You can enter up to 32 descriptive characters to identify this version and level of CMS, or you can press **ENTER** to accept the default version identifier, VM/XA CMS 5.6 *mm/dd/yy hh:mm*.

```
DMSINQ612R Enter installation heading:
```

The installation heading appears at the beginning of each output file created using this CMS nucleus.

■ or *heading* ■

You can enter up to 64 descriptive characters to serve as an installation heading, or you can press **ENTER** to accept the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM.

End of CMS Nucleus Generation Prompts and Responses

```
DMSINS327I The installation saved segment could not be loaded
```

Informational message: The CMSINST installation segment is not loaded and saved until you complete Step 27 on page 192.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

This is the default version identification. If you defined your own version identification, it appears here and each time that you IPL 190 or IPL CMS.

```
⋮
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

Step 16. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to MAINT's virtual printer. Ready the printer to send the CMS load map to MAINT's virtual reader:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt * ■
```

```
RDR FILE fileno SENT FROM MAINT
PRT AS fileno RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The CMS load map has been sent to MAINT's virtual reader. If you do not get this message, it has already been sent there by a previous IPL 190.

2. Identify the reader file:

```
query rdr * all ■
```

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,300 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

3. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft d ■
File fn ft D received from MAINT at * sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month, (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

4. To print a copy of the CMS load map, issue:

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Examine the load map for errors:

```
xedit fn ft ■
⋮
qquit ■
```

6. Continue with the next step.

Step 17. Tailor the Sample HCPRIO, HCPSYS, HCPBOX Files and the Product Parameter File

In this step, you will:

- Edit the real I/O configuration and system definition sample files
- Tailor the HCPBOX sample file if you wish to change either the default logo that appears on your screen when you log on the system or the default logo that appears on separator pages
- Tailor the product parameter file only if you have a reason for changing the supplied CP nucleus generation information
- Assemble the real I/O configuration file and the system definition files.

The time to perform this step varies depending upon the time you spend editing files.

Overview

Note: If you are applying updates with AUX files, follow the service procedures in Part 2, "Servicing the System."

As shipped on the product tape, the product parameter file and the sample files contain sample information and default parameters. Based on the requirements that were established in pre-installation planning, **you must examine and modify these files** to define your unique system configuration.

The product parameter file, 56643089 $PPF, contains information that the system generation tool (VMFBLD EXEC) uses when building system nuclei.

The sample files, which you loaded from the product tape to MAINT 295 in "Step 5. Invoke the ITASK EXEC" on page 126, contain the following information:

- The real I/O configuration file (HCPRIO ASSEMBLE) defines the configuration of your system input/output devices.
- The CP system control file (HCPSYS ASSEMBLE) describes the system residence device (XASRES) and various system parameters.
- The HCPBOX ASSEMBLE file defines both the logo that appears on your screen when you log on your system and the logo that appears on separator pages.

Note: You may find it useful to print copies of the system files for future reference. These include HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and your user directory (default USER DIRECT).

To tailor any of these files, use the VM/SP System Product Editor (XEDIT). XEDIT offers full-screen editing on display terminals.

Note: When you XEDIT ASSEMBLE files, issue SET TRUNC 72 so that you do not lose the continuation character in column 72.

Procedure

1. Access 295 as D:

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. See Appendix C, "VM/XA System Product Starter System Information" on page 725 for the sample files that correspond to your VM/XA System Product starter system, and "A Sample Product Parameter File" on page 384 for a sample product parameter file. Unless you alter them, the ITASK EXEC will use these files as they appear in Appendix C, "VM/XA System Product Starter System Information" when it creates the new CP nucleus. If you wish to alter the real I/O configuration file, the system definition files, or the product parameter file, follow the instructions in this step.

HCPBOX ASSEMBLE

If you want to change the design or contents of either the logo that appears when you log on to the system, or the logo that appears on separator pages, tailor the sample HCPBOX ASSEMBLE file. **Do not edit HCPBOX ASSEMBLE.** IBM services this file and bases the service on the version supplied. Use the procedure for local modifications.

1. Establish the appropriate minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you will receive a return code of 4 and a DMSWSU1900W message. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
vmfsetup 56643089 cp (all ■
erase 56643089 $setup a ■
```

2. Verify that HCPXA CNTRL lists an auxiliary control file for local changes:

```
xedit hcpxa cntrl ■
TEXT MACS HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT  AUXPAT TX$  * LOCAL PATCHES
LCL  AUXLCL LCL  * LOCAL MODIFICATIONS
TEXT AUXXA       *
qquit ■
```

HCPXA CNTRL should have a line like this one. The second field is the filetype of the auxiliary control file that points to local modifications.

3. Edit the auxiliary control file:

```
xedit hcpbox auxlcl ■
input ■
```

You will add your local modifications at the top of the file.

```
R00001LC LCL LC00001  *UPDATE HCPBOX LOGO ■
```

In this example. **R00001LC** is the filetype of the update file, and **LC00001** is the local tracking number. The rest of the line, beginning with the asterisk (*), is a comment explaining the purpose of this modification. The text deck created when you assemble HCPBOX with your local modifications will be called HCPBOX LCL00001.

```
■
file = = fm ■
```

Press **ENTER** to return to the command line.

fm is the CP LOCAL1 disk (295).

4. Create the update file and apply the updates:

```
xedit hcpbox assemble (ctl hcpxa ■
```

Instead of editing HCPBOX ASSEMBLE directly, you will apply changes to a copy of it.

```
DMSXUP178I Applying HCPBOX AnnnnnHP
```

The updates supplied by IBM for HCPBOX ASSEMBLE are being applied.

```
DMSXUP180W Missing PTF file HCPBOX R00001LC F5
```

Your update file cannot be found because you have not created it yet.

```
 HCPBOX R00001LC A1
      |...+....1....+....2....+....3....+
===== * * * Top of File * * *
=====    .
=====    .
=====    .

====>
```

The system copies the source file and allows you to edit the copy.

```
set serial off ■
set trunc 72 ■
⋮
file = = fm ■
```

You want the source file's sequence numbers and continuation characters to remain unchanged.

Make your changes and file the update file on the CP LOCAL1 (295) disk.

HCPRIO ASSEMBLE

You can use a procedure similar to the one shown for updating HCPBOX ASSEMBLE to update HCPRIO ASSEMBLE, or you can use the procedure shown below.

1. Edit the real I/O configuration source file. This file is on MAINT's 295 minidisk. The file is called HCPRIO ASSEMBLE.

   Use the XEDIT command to edit the file. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   You **must** make sure that your system volumes and consoles are defined properly in HCPRIO. If they are not, you will not be able to IPL your new system.

   For the real I/O configuration file, you may want to:

   - Delete those RDEVICE macro instructions for device numbers that the system will not be likely to use
   - Add RDEVICE macro instructions for device numbers that do not appear in the sample file.

Tailor HCPRIO ASSEMBLE to match the IOCDS, which is the data set that defines the I/O configuration for the hardware. See "System Installation on 308x, 4381, and 3090 Processor Complexes" on page 5.

Note: When tailoring HCPRIO, minidisk caching is not allowed for shared DASD. To control sharing on a device level, use the RDEVICE SHARED=YES option in HCPRIO or use the SET SHARED ON FOR *rdev* command. These will effectively turn off minidisk caching on a device basis. We recommend that you set the RDEVICE macro in HCPRIO rather than using the SET SHARED ON FOR *rdev* command.

See *VM/XA SP Planning and Administration* for details about HCPRIO macro instructions and for a high-level explanation of HCPRIO ASSEMBLE.

2. File the edited version of the real I/O configuration file on MAINT's 295 minidisk. Use the XEDIT subcommand FILE.

HCPSYS ASSEMBLE

You can use a procedure similar to the one shown for updating HCPBOX ASSEMBLE to update HCPSYS ASSEMBLE, or you can use the procedure shown below.

1. Edit the system definition source file. The file, called HCPSYS ASSEMBLE, is on MAINT's 295 minidisk.

   Use the XEDIT command to edit the file. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   For the system definition file, you may want to:

   - Change the device number for the system residence device on the SYSRES macro instruction.

     Note: By convention, IBM assigns device number 0123 for the system residence device. This device number appears in four places:

     - On the DIRECTORY control statement in the user directory.
     - On an MDISK control statement in MAINT's entry in the user directory (this gives MAINT read/write access to the system residence device).
     - On the SYSRES macro instruction.
     - In the product parameter file. (The device number for the system residence device might not be included in the product parameter file [56643089 $PPF].)

     If you change the device number in one place, be sure to make an identical change in the other three places.

   - Modify the SYSSTORE macro instruction. Set the RMSIZE operand to the storage of your processor. If your system will have a virtual=real area (including storage for virtual=fixed guests), set the VRSIZE operand. If any virtual machines will need to issue DIAGNOSE X'98' in System/370 mode, or in 370-XA mode with 24-bit addressing, set the RIO370 operand.
   - Modify the SYSTIME macro instruction for your time zone.
   - Tailor the volume identifiers on the SYSCPVOL and SYSUVOL macro instructions to match the volume identifiers your system will use. Because this action eliminates extraneous warning messages, the action helps the real system operator identify volumes that are not mounted at IPL time.
   - Supply your own system identifier on the SYSID macro instruction.

   See *VM/XA SP Release Guide* for Release 2.1 for details on HCPSYS macro instructions.

2. File the edited version of the system definition file on MAINT's 295 minidisk. Use the XEDIT subcommand FILE.

56643089 $PPF

1. The default 56643089 $PPF should be appropriate for most installations, but if your experience suggests changes, you can override 56643089 $PPF.

   DO NOT alter this file unless you have a special need.

   The best way to change the product parameter file is to create an override file. For a discussion of override files, see "The Product Parameter Override File" on page 402.

   Use the XEDIT command to create an override file on the TASK (5E5) minidisk. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

   Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC must all match. If you change an address in any one of these places, you must change it in all the others. We recommend that you do not change any addresses.

2. File the override file on the TASK (5E5) minidisk. Use the XEDIT subcommand FILE.

Assemble the Files

1. Assemble the HCPRIO, HCPSYS, and HCPBOX ASSEMBLE files:

```
itask assemble filename ■
:
DMSWHM1907I Assembling filename
```

If an error occurs while assembling any of the system definition files, see "Correcting Assembly Errors."

```
DMSUPD178I Updating HCPBOX ASSEMBLE B1
DMSUPD178I Applying HCPBOX AnnnnnHP F5
DMSUPD178I Applying HCPBOX ft B5
DMSWHM1909I filename {TEXT|ft} A created
RDR FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

You will see these messages only when you assemble HCPBOX ASSEMBLE, and only if you updated it.

2. Copy these text files from your A-disk to the CP LOCAL1 (295) minidisk:

   HCPSYS *ft*
   HCPRIO *ft*
   HCPBOX *ft*

```
copy filename {text|ft} a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase filename {text|ft} a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the CP LOCAL1 minidisk (the default is 295).

Correcting Assembly Errors

To correct assembly errors in system definition ASSEMBLE files, do the following:

1. Examine the flagged statements. See *VM/XA SP Planning and Administration* for directions on coding.
2. Correct the statements causing an error by editing the system definition file with the System Product Editor (XEDIT).
3. Reassemble the system definition file in which the change was made using the procedure in "Assemble the Files."
4. Repeat the sequence until all files assemble correctly.

Step 18. Generate the New CP Nucleus

In this step, you will use the ITASK EXEC to build your new CP nucleus.

This step takes approximately 20 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Purge all unnecessary files from your reader to increase available spool space.
2. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the ITASK EXEC to build the CP nucleus:

```
itask build cp noassem ■
```

Use the **noassem** only if you have already assembled HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and HCPBOX ASSEMBLE. If you have not assembled these files, you **must** omit the **noassem** operand, so that ITASK will assemble them now.

If you assemble the files here, copy the text decks from MAINT 191 to the CP LOCAL1 minidisk (the default is 295), then erase them on MAINT 191.

```
nnnnnnnn FILES CHANGED
DMSACC724I 191 replaces A(191)
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

RDR FILE fileno TO MAINT COPY 001 NOHOLD
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The CP nucleus is being created. It will be sent to your reader.

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CP, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

4. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

5. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 10 on page 165. It has approximately 75,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

6. Order your reader so that the CP nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CP nucleus.

7. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED  NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

9. If necessary, redefine your virtual storage. First, find out how much virtual storage MAINT has:

```
query virtual storage ■
STORAGE = nnnnM
Ready; T=n.nn/n.nn hh:mm:ss
```

Now determine how much storage you need:

- You need at least 16 megabytes to IPL your virtual reader in substep 10 on page 165.
- If you are generating a CP nucleus with a preferred virtual machine, MAINT's virtual storage must be at least 4 megabytes greater than the sum of the VRSIZE, VRFREE, and RIO370 operands in the SYSSTORE macro instruction in HCPSYS ASSEMBLE. For more information on the SYSSTORE macro instruction, refer to *VM/XA SP Planning and Administration*.

  If MAINT's virtual storage must be set to a value greater than 32MB (the maximum size for MAINT as defined by the sample USER DIRECT file), then you must tailor the directory (default USER DIRECT) to change the maximum virtual machine size for MAINT. Once the directory is changed, bring it online by issuing DIRECTXA *directory-name*. If this is not done, you receive wait state 8028 when the reader is IPLed.

If you need to increase your storage to more than 16 megabytes, first issue:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

Storage addresses greater than 16M require 370-XA architecture. If your machine is already in 370-XA mode, you will not get a response.

If you need to increase your storage (even if you need only 16M), issue:

```
define storage nnnnm ■
STORAGE = nnnnM
STORAGE CLEARED - SYSTEM RESET
```

10. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

```
hh:mm:ss * MSG FROM MAINT   :
  HCPGEN9010W NUCLEUS LOADED ON XASRES
  HCPGIR450W  CP ENTERED; DISABLED WAIT
  PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus was successfully loaded on the system residence device.

If you receive disabled wait state code 8028, see "Correcting IPL Errors" below.

If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

11. Issue the following commands to ensure that these settings are in effect:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
```

If your machine is already in 370 mode, or if your storage is already 16M, you will not get a response.

You have now loaded the CP nucleus.

Correcting IPL Errors

If you receive disabled wait state code 8028 when you try to IPL your reader, you must correct the problem as follows:

1. Close the reader. Issue CLOSE RDR.
2. Determine the reader file number of the IPL deck. Issue QUERY RDR ALL.
3. Take the IPL deck out of USER HOLD status. Issue CHANGE RDR *nnnn* NOHOLD, where *nnnn* is the reader file number of the IPL deck. If this step is not done, you will receive wait state 232 when the reader is re-IPLed.
4. Tailor the directory (default USER DIRECT) as described in substep 9 on page 164.
5. Reissue DIRECTXA *directory-name*.
6. IPL the reader. Issue IPL C CLEAR.

Correcting Load Errors

If the new CP nucleus fails to load, do the following:

1. IPL CMS:

```
ipl 190 clear ■
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If you have changed the version heading, your own heading will appear.

2. Access 295, then inspect the CP load map and virtual PSW. An error in loading may show up in either the load map or in the PSW.

   For a list of loader wait-state codes, refer to *VM/XA SP System Messages and Codes Reference*.

3. Correct the error.

4. Go back to the point where the error may have occurred. If you cannot determine where the error may have occurred, see "Step 13. Install Assembler H Version 2 Program Product" on page 146.

Step 19. Save and Print the CP Load Map

In this step, you will:

- IPL CMS
- Save the load map of the new CP system
- Print a copy of the new CP load map.

This step takes approximately 10 minutes.

1. When you built CP, the CP load map was spooled to MAINT's virtual printer. The load map must be transferred to MAINT's virtual reader. If you have IPLed 190 since building CP, the load map may have already been spooled from the virtual printer to the virtual reader.

   Query the printer to see if the load map is still there. If it is, note the file number. If it is not, skip substep 3 and go to substep 2.

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
MAINT  fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

   The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

2. IPL CMS:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

CP Load Map in Virtual Printer Only

3. Transfer the printer file to your reader:

```
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

End of CP Load Map in Virtual Printer Only

4. Access the CP LOCAL1 disk:

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

   This is the disk on which you will receive the load map.

5. Identify the reader file:

```
query rdr * all ■

ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST    USERFORM OPERFORM KEEP MSG
MAINT    fileno A PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG STANDARD STANDARD OFF  OFF
```

The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

6. Bring in the file from the reader to the CP LOCAL1 disk:

```
receive fileno fn ft d ■
fn ft D1 created
DMSRDC738I Record length is 132 bytes
File fn ft D received from MAINT at *
  sent as (none) (none) D
```

This command saves the load map on the CP LOCAL1 (295) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1−9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

7. To print a copy of the load map for the CP nucleus, issue:

```
spool 00c class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The load map is printed on your system printer.

8. Examine the load map for errors:

```
xedit fn ft ■
⋮
```

9. Examine the load map for unresolved symbols. There should be no unresolved symbols.

```
locate /unresol ■

DMSXDC546E Target not found
qquit ■
```

You are still in XEDIT mode.

Step 20. Load the New CP Nucleus

In this step, you will:

- Shut down the VM/XA System Product starter system
- Perform an initial program load of the new CP nucleus.

This step takes approximately 10 minutes.

1. Shut down the starter system you are now using. A class A user (usually the primary system operator) must issue the SHUTDOWN command.

```
shutdown ■
SYSTEM SHUTDOWN STARTED
Ready; T=n.nn/n.nn hh:mm:ss
SYSTEM WARM START DATA SAVED
SYSTEM SHUTDOWN COMPLETE
HCPGIR450W CP ENTERED; DISABLED WAIT
            PSW 000A0000 00000961
```

2. IPL the virtual address of your system residence device.

   If you are unable to IPL your system residence device, go back to "Step 2. Restore the VM/XA System Product Starter System to Disk" on page 115 and continue from there.

```
ipl vdevno clear ■
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;
SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss,
LOADED FROM XASRES
*****************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
*****************************************
⋮
HCPISU951I CP VOLID volid NOT MOUNTED
```

Note: A 9025 disabled wait state indicates that your system volume (XASRES) is not defined in the HCPRIO ASSEMBLE file. A 1010 disabled wait state indicates that your console is not defined in the HCPRIO ASSEMBLE file.

3. During the initialization phase, respond **cold drain** to the START message:

```
START ((COLD|WARM|FORCE) (DRAIN) (DISABLE) (NODIRECT))|(SHUTDOWN)
cold drain ■
```

```
NOW hh:mm:ss EDT weekday mm/dd/yy
CHANGE TOD CLOCK (YES|NO):
no ■
```

You cannot set the clock at second level. Answer **no**.

```
The directory on volume XASRES at address nnnn
   has been brought online.
There is no logmsg data
FILES: NO RDR, NO PRT, NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy
```

```
HCPCPJ951I CP volid volid not mounted
⋮
```

volid is a DASD volume listed on the SYSCPVOL macro instruction of HCPSYS. If the volume is not one that you are using, ignore the message.

You are now logged on to the OPERATOR user ID.

```
STORAGE = 0016M
FILES 0000001 RDR, 0000001 PRT, NO PUN
```

```
XAUTOLOG AUTOLOG1
HCPAUTO53E AUTOLOG1 not in CP directory
XAUTOLOG EREP
HCPAUTO53E EREP not in CP directory
XAUTOLOG DISKACNT
HCPAUTO53E DISKACNT not in CP directory
```

After you have modified the user directory, these error messages will no longer appear.

4. Define the storage that you want for the OPERATOR user ID:

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
```

This is an example. You may define more than 16M.

5. Enable the devices in your revised HCPRIO file:

```
enable all ■
hh:mm:ss Command complete
terminal mode vm ■
```

6. IPL 190:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

```
DMSACP1123E A(191) device error
Ready; T=n.nn/n.nn hh:mm:ss
```

If you see this error message, you have not formatted the operator's 191 minidisk.

7. Format the operator's 191 minidisk **only if you have not already done so:**

```
format 191 a ■
DMSFOR603R FORMAT will erase all files on disk A(191).
           Do you wish to continue?
           Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
opr191 ■
DMSFOR733I Formatting disk A
DMSFOR732I n cylinders formatted on A(191)
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Disconnect from OPERATOR and log on as MAINT.

   Note: If you do not have XAP001, you will receive this message:

   ```
   HCPLND108E MAINT 0124 NOT LINKED
   ```

   when you log on as MAINT. If you are using 3380 or 3390 DASD, you can ignore this message.

Step 21. Update the User Directory

In this step, you will:

- Edit the sample user directory
- Use the DIRECTXA command to update the user directory.

The time to perform this step varies depending on the time you take to edit files.

1. See Appendix C, "VM/XA System Product Starter System Information" on page 725 for sample user directories.

   Note: All logon passwords are NOLOG and there are no MDISK passwords, except for some of MAINT's minidisks. You may want to change these.

2. Edit the sample user directory (USER DIRECT on MAINT's 295 minidisk):

```
access 295 d ■
xedit user direct ■
⋮
```

   Be sure to change the logon passwords in USER DIRECT from NOLOG to your own non-restricted installation passwords. (For a list of restricted passwords, see Appendix F, "Restricted Logon Passwords" on page 859.) Also, be sure to change the logon password you provided for the MAINT and OPERATOR virtual machines in "Step 5. Invoke the ITASK EXEC" on page 126. This password appeared in the console file.

   Also, take this opportunity to add or change MDISK passwords for the MDISKs defined in USER DIRECT. Note that the read, write, and multiple-write passwords are positional. Refer to the skeletal MDISK statement provided in USER DIRECT. If you will be starting accounting and error recording later, this is a good time to add or change MDISK passwords for the DISKACNT and EREP 191 minidisks.

   For more information about the user directory, see *VM/XA SP Planning and Administration.*

3. When your changes are complete, enter **file** on the command line.

4. Use the DIRECTXA command to update and place the user directory online:

```
directxa user direct ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ON LINE
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 22. ITASK Loads HELP Files from the Product Tape (Volume 2)

In this step, the ITASK EXEC loads HELP files from Volume 2 of the product tape.

The time this step takes varies from system to system, but a good estimate is 40 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 2 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD AMENGHLP, LOAD UCENGHLP, or LOAD HELP operands to load the HELP files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK command again.

```
itask load {amenghlp|ucenghlp|help} ■
```

The HELP files on the tape are in mixed-case American English:

- Use the **amenghlp** operand to load the HELP files to the 19D disk.
- Use the **ucenghlp** operand to load the HELP files directly to the 19C disk, then convert them to uppercase. Conversion takes about 20 minutes.
- Use the **help** operand to load the HELP files to the 19D disk, then copy them to the 19C disk and convert the copied files to uppercase.

```
DMSWSL409I Loading AMENGHLP FILES to the
           19D disk attached to MAINT
```

You will see this message if you used the **amenghlp** or **help** operand.

```
DMSWSL409I Loading UCENGHLP FILES to the
           19C disk attached to MAINT
```

You will see this message if you used the **ucenghlp** operand.

```
CONVERTING HELP FILES TO UPPERCASE
Ready; T=n.nn/n.nn hh:mm:ss
```

You will see this message if you used the **ucenghlp** or **help** operand.

Step 23. ITASK Loads Source Files from the Product Tape (Volume 3)

In this step, the ITASK EXEC loads CP, CMS, and dump viewing facility source files from Volume 3 of the product tape.

The time this step takes varies from system to system, but a good estimate is 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 3 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD ALL3 operands to load all the source files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK LOAD ALL3 command again.

```
itask load all3 ■

DMSWSL409I Loading CP SOURCE to the
           394 disk attached to MAINT
DMSWSL409I Loading CMS SOURCE to the
           393 disk attached to MAINT
DMSWSL409I Loading DUMPVIEW SOURCE to the
           393 disk attached to MAINT

Ready; T=n.nn/n.nn hh:mm:ss
```

These are messages from ITASK as it loads files from the product tape.

ITASK has finished restoring source files from Volume 3 of the product tape.

Note: Source files are **not** unpacked during the load.

Step 24. Install Environmental Record Editing and Printing

In this step, you will:

- Read the documentation for the Environmental Record Editing and Printing (EREP) Program
- Follow the directions to install EREP.

The time to perform this step may vary, but a good estimate is 15 minutes.

Note: IBM packages the Environmental Record Editing and Printing (EREP) Program, VM Feature, 5654-260, as a separate program product. This program product has its own installation procedure.

1. Read the documentation for EREP:
   - The following memos in tape files 1 and 2 of the VMSUP EREP tape:
     - 5654260B MEMO
     - F5654260 MEMO2
     - I5654260 MEMO
     - 5654260B SERVICE
   - The Program Directory for EREP, which describes the installation procedure for EREP.

     Note: This document does not come with the VMSUP tape. You must order it separately.
   - The discussion of EREP in *VM/XA SP Planning and Administration.*
2. Follow the directions in the EREP *Program Directory* to install EREP from the VMSUP EREP tape.

   Note: You will receive messages HCPCRC8059I and HCPCRC8060I until EREP is installed.
3. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 25. Install the Printer Image Library

In this step, you will install an image library for your printer and save it on tape.

This step takes approximately 10 minutes.

Warning: Do not attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.

1. Load (IPL) 190. This is the CMS system disk.

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you receive a return code of 4 and a DMSWSU1900W message. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
access 5E5 b ■
vmfsetup 56643089 cp ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the IMAGELIB command. The command requires the filename of the image library (*fn* CNTRL). IBM supplies default image library control files called IMAG*xxxx* CNTRL, where *xxxx* is the printer number: 3800, 1403, 3203, 3211, 3262, 4245, or 4248.

   For more information about image libraries, see *VM/XA SP Planning and Administration.*

```
imagelib libname ■                         If you use the IBM-supplied image library,
HCPNMT247I NAMED IMAGE IMAGxxxx CREATED    libname is IMAGxxxx, where xxxx is the printer
Ready; T=n.nn/n.nn hh:mm:ss                number.
```

4. Check the status of the image library:

```
query img all ■
```

```
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*IMG     nnnn IMG  A  nnnn mm/dd hh:mm:ss fn       IMG      MAINT
```

Check the current state of the file (CL), the filename, and filetype.

5. Issue the appropriate LOADBUF command and start the printer. See *VM/XA SP CP Command Reference* for more information.
6. Mount a scratch tape with ring on an available 3420, 3422, or 3430 tape drive, or insert a cartridge into an available 3480 unit, with the thumbwheel turned until the white dot is not visible. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.
   - The tape is suitable for a 3420, 3422, 3430, or 3480 tape unit.

   Note: This procedure assumes the tape mode is 1600 bits per inch.
7. Issue the SPTAPE command to dump the image library to tape:

```
sptape dump vdevno img all run ■
```

Substitute the virtual device number of the tape drive for the value *vdevno*. The operand RUN specifies that SPTAPE rewinds and unloads the tape after the operation.

```
DUMPING vdevno IMG *IMG    nnnn A
:
SPTAPE DUMP FUNCTION ON
   DRIVE vdevno COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The message from SPTAPE tells you the file is being dumped to tape.

8. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE LOAD 181 IMG ALL RUN command to restore the image libraries. For more information about the SPTAPE command, see *VM/XA SP Real System Operation.*
9. If necessary, reattach your tape drive.

Step 26. Load, Build, and Save GCS

You must install the GCS (group control system) component if you plan to install SNA products or RSCS Version 2. **Before you begin, read "Planning for the Group Control System (GCS)" on page 9.**

If you do not want to install GCS, go to "Step 28. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments" on page 196.

To install GCS, you will do the following steps:

- Load GCS from Volume 2 of the product tape.
- Invoke the GROUP EXEC and complete a series of panels to create a GCS configuration file.

  Note: If you do not have a full-screen display device, you cannot use the GROUP EXEC because you cannot display the panels. In that case, you must build the configuration file manually, using the build macros described in the *VM/XA SP Group Control System Command and Macro Reference.*
- After you complete the panels, you invoke the ITASK EXEC with the BUILD GCS parameters.

  ITASK:
  - Modifies a copy of the GCS loadlist (GCSLOAD EXEC) and changes the entry that contains the filename of the GCS configuration file (the default is GCS) to match the filename of the configuration file that you just created (the default is also GCS).
  - Files the modified loadlist on the MAINT 295 minidisk. The modified loadlist is used during the generation of the GCS named saved system.
  - Renames the filetype of the GCS configuration file from GROUP to ASSEMBLE.
  - Invokes VMFHASM EXEC to assemble the configuration file.
  - Invokes VMFBLD EXEC with the BUILD GCS parameters to build and save the GCS nucleus.
- After you build and save the nucleus, you can save and print the GCS load map, which contains the storage addresses of the nucleus control sections and entry points.
- If you want to install more than one GCS nucleus, you must re-access the GCS minidisk structure and re-invoke the GROUP EXEC to create another GCS configuration file. Each configuration file must have a unique name and have an entry in the SAMPNSS EXEC.

This step takes approximately 30 minutes for each GCS system.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC must all match. If you change an address in any one of these places, you must change it in all the others. We recommend that you do not change any addresses.

1. Attach a tape drive to MAINT.
2. Mount Volume 2 of the product tape.

3. Issue ITASK LOAD GCS to load the GCS code:

```
itask load gcs ■
DMSWSL409I LOADING GCS INTERFACE TO
     THE 190 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS OBJECT TO
     THE 595 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTDELTA TO
     THE 596 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORDELTA TO
     THE 596 DISK ATTACHED TO MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

4. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Establish the correct minidisk order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Set up the GROUP EXEC messages in storage:

   Notes:

   a. If you are REBUILDING a GCS nucleus and your current configuration file does not require changes, you can skip this part of the step and go directly to the nucleus build operation on page 187.

   b. If you do not have a full-screen display device, you **cannot** use the GROUP EXEC to build the configuration file. **Do Not** set up the GCS messages in storage. Instead, refer to the *VM/XA SP Group Control System Command and Macro Reference* and use the build macros described there to build the configuration file manually. Then continue this step with the nucleus build operation on page 187.

```
copyfile csimes[y] [text|txtnnnnn] fm csiume[y]text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

y is the country code for your system default national language: blank for mixed-case American English and **b** for uppercase American English. You can use QUERY LANGLIST to determine your system default national language.

The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a UM*nnnnn* field, where *nnnnn* is the PTF number.

Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*. You will have one with a filetype of TEXT.

This command creates a temporary GCS message file (containing the messages for the GROUP EXEC) that has the filename required by the SET LANGUAGE command. A permanent message file will be created when you generate the GCS nucleus. If there has been any service to the message file, the permanent file will reflect it.

```
set language langid (add csi user ■
Ready; T=n.nn/n.nn hh:mm:ss
```

langid identifies your system default national language. At this point, *langid* will be either **ameng** (mixed-case American English) or **uceng** (uppercase American English).

This command sets the language of the temporary GCS message file and places the file in user storage.

7. Invoke the GROUP EXEC to display the configuration panels.

   Use your installation reference worksheet to help you complete the panels. For more information, see "Planning for the Group Control System (GCS)" on page 9.

```
group systemname ■
```

This command assigns *systemname* as the filename of the GCS configuration file that you are creating and invokes the Primary Option Menu. The *systemname* parameter is optional at this time. If specified, it must match the system name in the DEFSYS entry for this GCS system in the SAMPNSS EXEC (the sample system name is GCS). If you specify a system name here, the Primary Option Menu appears with the system name filled in. If you do not specify a system name here, you must specify one on the Primary Option Menu.

Table 3. Function Keys Used with the GROUP EXEC Panels	
Key	**Function**
PF1 HELP	Shows information about the panel you are looking at.
PF2 CLEAR	Clears the input areas where you enter information.
PF3 END	Leaves the present panel and returns you to a previous one. • If you press **PF3** on the Primary Option Menu, you return to CMS. • If you press **PF3** on any other screen, you return to the Primary Option Menu. • If you press **PF3** after you have entered information, but have not saved it, you receive the message: `577E File has been changed; type QQUIT to quit anyway`
PF5 REFRESH	Fills in the panel's input areas with the values you last saved there.
PF6 SAVE	Saves information you've entered on the panel (for the configuration file *systemname* GROUP).
PF7 PREVIOUS	Returns to the previous panel (if any).
PF8 NEXT	Moves ahead to the next panel (if any).
PF9 VERIFY	This PF key is not supported by VM/XA SP.
PF12 CURSOR	Moves the cursor to the panel's command line.
PF4, **PF10**, **PF11**	Not used.
ENTER	Processes any valid CP or CMS command typed on the command line. Two specific commands you can enter are: 1. QQUIT, entered from the Primary Option Menu, returns you to CMS. If entered on any other panel, it returns you to the Primary Option Menu. 2. CANCEL, entered on any panel, returns you to CMS.

```
 GRP1                  GCS GROUP - PRIMARY OPTION MENU                  Primary
-------------------------------------------------------------------------------

Fill in the blanks with the required information and then press the ENTER key.

       Type/change the name of the saved system that is being defined.

                        SYSTEM NAME : ________

          Type one number from the list below to display/update the:

                     1.  Authorized VM Userids.
                     2.  Saved System Information.
                     3.  Saved Segment Links.

             Type your choice here: _


-------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  ...      6  ...
  PF: 7  ...      8  ...      9  VERIFY   10  ...    11  ...     12  CURSOR

===>
```

8. To complete the Primary Option Menu:

   a. **Fill in or change the SYSTEM NAME.**

      If you invoked GROUP with the *systemname* parameter, this panel appears with the SYSTEM NAME already filled in.

   b. **Select the next panel (1, 2, or 3) and press ENTER.**

      The first time you leave the Primary Option Menu, select panel 1; the next time, panel 2; the third time, panel 3.

```
GRP11          AUTHORIZED VM USERIDS FOR <      >       PAGE 1 OF 1
------------------------------------------------------------------------------



      To ADD,    fill in the blanks with the authorized VM userids.
      To CHANGE, type a new userid over the userid to be changed.
      To DELETE, type blanks on the line.

         Pressing the ENTER key or PF6 key will save the update.


       ________   ________   ________   ________________
       ________   ________   ________   ________________
       ________   ________   ________   ________________
       ________   ________   ________   ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY  10  ...    11  ...      12  CURSOR

====>
```

9. To complete the Authorized Userids panel:

   a. **Type in the user IDs.**

      Press **ENTER** or **PF6** after typing the user IDs to save your information. Every time you press **ENTER** or **PF6**, the GROUP EXEC tells you how many user IDs it has processed. Use **PF7** and **PF8** if you have more than one screenful of user IDs.

   b. **Return to the Primary Option Menu.**

      Press **PF3**. If you forget to press **ENTER** or **PF6**, you will remain on this screen and see the message:

      ```
      577E File has been changed; type QQUIT to quit anyway
      ```

      To continue, simply press **PF6**. You will receive a message telling you how many authorized user IDs the EXEC has processed. Press **PF3** again to return to the Primary Option Menu.

```
GRP121            SAVED SYSTEM INFORMATION FOR  <     >       PAGE 1 OF 2
------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.

   SYSTEM DISK address (S-disk) . . . . .:        595
   SYSTEM DISK EXTENSION address (Y-disk):        59E

   USERID to RECEIVE STORAGE DUMPS. . . .:        ________
   RECOVERY MACHINE USERID (required) . .:        ________

   GCS TRACE TABLE SIZE (minimum 4K). . .:        ___16K


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  ...      8  NEXT     9  VERIFY  10  ...    11  ...      12  CURSOR

====>
```

10. To complete the first Saved System Information panel:

    a. **Specify your disk addresses.**

       The default 595 and 59E virtual addresses are already saved and verified. If you must choose different addresses, type over the default values.

       Note: Make sure the new information is correct; the GROUP EXEC does not prevent you from saving invalid information.

    b. **Name an authorized dump receiver.**
       Name an authorized recovery machine.

       GCS requires you to enter a recovery machine user ID. The GROUP EXEC tells you whether that user ID is valid, but it does not prevent you from saving an invalid entry.

    c. **Specify a trace table size.**

       By default, the GROUP EXEC saves a value of 16K. If you decide to save a different amount, simply type the new value over the default value.

    d. **Save your input.**

       If you are satisfied with your choices, press **PF6** or **ENTER** to save the information.

       Warning: If you do not save the information before going on, the system will not accept it as input.

    e. **Go to the second panel.**

       Press **PF8** to continue on the second Saved System Information panel.

```
GRP122            SAVED SYSTEM INFORMATION FOR <      >        PAGE 2 OF 2
-------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.


MAXIMUM NUMBER of VIRTUAL MACHINES (required). .:_____________________________

SYSTEM ID (maximum 130 characters) . . . . . .:_____________________________

_______________________________________________________________________________

_______________________________________________________________________________



-------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...      5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  ...      9  ...     10  ...     11  ...      12  CURSOR

====>
```

11. To complete the second Saved System Information panel:

    a. **Specify a MAXIMUM NUMBER.**

       Simply type a value of 1 or more, and press **ENTER**. The GROUP EXEC responds with a message *only* if you enter an invalid value.

    b. **Type in your SYSTEM ID text.**

       This is optional. Move the cursor to the SYSTEM ID space (if it is not already there) and enter your text. The GROUP EXEC does not let you enter any more than 130 characters.

    c. **Save your input and return to the Primary Option Menu.**

       Press **ENTER** to save your information and then press **PF3**.

```
GRP13        AUTOMATIC SAVED SEGMENT LINKS FOR <      >    PAGE 1 OF 1
------------------------------------------------------------------------------


      To ADD,    fill in the blanks with the saved segment names
                 that will be linked automatically during
                 initialization of this virtual machine group.
      To CHANGE, type a new saved segment name over the saved
                 segment name to be changed.
      To DELETE, type blanks on the line.

         Pressing the ENTER key or PF6 key will save the update.

         ________   ________   ________   ________________
         ________   ________   ________   ________________
         ________   ________   ________   ________________
         ________   ________   ________   ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5   REFRESH 6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY  10  ...    11   ...    12  CURSOR

====>
```

12. To complete the Automatic Saved Segment Links panel:

    Note: Do not include CMSVSAM and CMSBAM segments here.

    a. **Type in the segment names.**

       Press **ENTER** or **PF6** after typing each segment name to save your information and advance to the next space at the same time. Every time you press **ENTER** or **PF6**, the EXEC tells you how many segment names it has processed. Use **PF7** and **PF8** if you have more than one screen of names.

    b. **Return to the Primary Option Menu.**

       If you have not saved your input yet, press **ENTER** or **PF6**. Press **PF3** to return to the Primary Option Menu.

13. Once you have provided the necessary input on all panels and returned to the Primary Option Menu for the last time, press **PF3** to exit from the GROUP EXEC.

14. Remove the temporary GCS message file from user storage and erase it from your A-disk:

    ```
    set language langid (delete csi user ■
    erase csiume[y] text a ■
    ```

    langid is your system default national language. If you have not installed a different one, your default is **ameng**.

15. Spool punch output to your own virtual reader:

    ```
    spool punch * ■
    Ready; T=n.nn/n.nn hh:mm:ss
    ```

16. Invoke the ITASK EXEC with the BUILD GCS parameters:

```
itask build gcs systemname ■
```

systemname must match the name you specified when you issued the GROUP EXEC. If you do not specify a name, the default is GCS.

```
HCPNSD440I The Named Saved System (NSS) systemname
            was successfully defined
            in fileid fileno
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT

DMSUPD181E No update files were found
DMSWHM1907I Assembling systemname
DMSWHM1909I systemname TEXT A created
191 replaces A (191)
5E5 replaces B (5E5)
```

If You Receive an Assembly Error

If you receive an assembly error, you may have to reestablish the minidisk access order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■
```

Then go back and check your configuration file, which ITASK has renamed *systemname* ASSEMBLE. Refer to "Planning for the Group Control System (GCS)" on page 9 for information regarding required fields. Correct the file or go through the GROUP EXEC panels again to recreate the file. (If you recreate the configuration file, you must use the same *systemname*.) After you recreate the configuration file, rename or erase the old *systemname* ASSEMBLE file, then rename the new configuration file from GROUP to ASSEMBLE. Issue the ITASK BUILD GCS *systemname* command to assemble the configuration file and to build and save the nucleus.

End of If You Receive an Assembly Error

```
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
nnnnnnn FILES CHANGED
:
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

These are informational messages. You can ignore them.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The GCS nucleus has been sent to MAINT's virtual reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF which is now being applied has another PTF, outside GCS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

17. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

18. Copy the text deck, configuration file, ASSEMBLE file, and EXEC from MAINT 191 to MAINT 495:

```
access 495 d ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname text a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname text a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname group a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname group a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname assemble a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname assemble a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

19. Verify that the GCS nucleus is in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss fn  ft      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 22 on page 189. It has approximately 8,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

The name of the IPL deck is the same as the last entry in the GCS load list. If you have difficulty identifying the IPL deck and need to know the reader file number, refer to the last entry in the GSC load list *systemname* EXEC. (The default name is GCSLOAD EXEC).

20. Order your reader so that the GCS nucleus will be processed first:

```
order rdr fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the GCS nucleus.

21. Query the reader. If it is not class *, spool it as class *:

```
query virtual c ■
RDR 000C CL cl NOCONT NOHOLD EOF READY
    000C nnnnnCLOSED NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
spool c class * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

22. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■

MSG FROM MAINT  :  CSIINI134I systemname has nnnnnn
   bytes of available common free storage
HCPNSS440I Named Saved System (NSS) systemname
            was successfully saved in fileid fileno

PRT FILE fileno SENT FROM MAINT PRT WAS fileno
  RECS nnnn CPY 001 NOHOLD NOKEEP
```

This message indicates that the GCS load map file has been sent to MAINT's virtual printer. If you plan to save or print the GCS load map, record *fileno*.

23. IPL 190:

```
ipl 190 ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: If necessary, purge your reader to free spool space.

Saving and Printing the GCS Load Map

24. The GCS load map has been sent to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME  NAME TYPE   DIST
MAINT    fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

The GCS load map is the file with a blank filename and filetype. It has approximately 2,200 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

```
#cp transfer prt fileno to * rdr ■
access 495 d ■
receive fileno fn ft d ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print fn ft d ■
```

End of Saving and Printing the GCS Load Map

Installing Multiple GCS Systems

25. You can install more than one GCS system, if you defined the additional systems in the system directory.

    To install each additional system:

    a. Reestablish the minidisk access order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■
```

    b. Set up the GROUP EXEC messages in storage (full-screen display devices only):

```
copyfile csimes[y] [text|txtnnnnn] fm csiume[y]text a ■
```

    The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a UM*nnnnn* field, where *nnnnn* is the PTF number.

    Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*. You will have one with a filetype of TEXT.

```
set language langid (add csi user ■
```

    c. Create a configuration file for each nucleus.

    - If you are using a full-screen display device, issue:

```
group systemname ■
```

      systemname is a unique filename. Then complete the GCS panels to define the system parameters.

    - If you are **not** using a full-screen display device, refer to the *VM/XA SP Group Control System Command and Macro Reference*. Use the build macros to create the configuration file.

    d. Remove the temporary GCS message file from user storage and erase it from your A-disk (full-screen display devices only):

```
set language langid (delete csi user ■
erase csiume[y] text a ■
```

    e. For each new GCS system, add a DEFSYS statement to the SAMPNSS EXEC. For information on the DEFSYS command, see the *VM/XA SP CP Command Reference*.

f. For each new GCS system, add an override section at the end of the product parameter file. This example shows an override section for a GCS system called **NEWGCS**:

```
:NEWGCS. GCS
:BLD. REPLACE
 NEWGCS VMFBDNUC BUILD1
:END.
```

You must also add the name of the new GCS system to the :OVERLST. tag at the top of the product parameter file, for example:

```
:OVERLST. CORCP CORCMS CORGCS NEWGCS
```

g. For each GCS system, assemble the configuration file, build and save the new GCS nucleus, and copy the text deck, configuration file, ASSEMBLE file, and EXEC to the 495 disk:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■


itask build gcs systemname ■
access 495 d ■
copy systemname text a = = d (replace ■
erase systemname text a ■
copy systemname group a = = d (replace ■
erase systemname group a ■
copy systemname assemble a = = d (replace ■
erase systemname assemble a ■
ipl 00c clear ■
ipl cms ■
```

Before you build another GCS nucleus, save or print the GCS load map, as indicated above. Remember to give each load map file a different name.

End of Installing Multiple GCS Systems

26. Verify that all your GCS segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss GCS         NSS      MAINT
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss systemname  NSS      MAINT
⋮
```

27. When you complete your GCS installation, go to "Step 27. Save CMS" on page 192.

Step 27. Save CMS

In this step, you will save CMS as a named saved system in 370 mode and 370-XA mode, at default addresses.

This step takes approximately 10 minutes.

Note: For general information about CMS as a named saved system, see *VM/XA SP Planning and Administration.*

1. Make sure:
   - You are logged on as MAINT.
   - You have finished loading all files to the 190 and 19E minidisks. After this step, any change to files on these disks requires resaving CMS.
   - You have 16 megabytes of virtual storage. To determine your storage size, issue QUERY VIRTUAL STORAGE. If you have less than 16 megabytes, issue DEFINE STORAGE 16M.
   - CMS is running. If not, load CMS by issuing IPL 190 CLEAR. Wait for the CMS version identification to appear on the screen. Then press **ENTER**.
2. Invoke the SAMPNSS EXEC:

```
sampnss cms cmsxa ■
```

The SAMPNSS EXEC issues the DEFSYS command.

Note: If you use "CMS" (the default name) for the System/370 name, users will receive the new level of CMS each time they issue the command IPL CMS.

```
HCPNSD440I Named Saved System (NSS) CMS was
           successfully defined in
           fileid fileno

HCPNSD440I Named Saved System (NSS) CMSXA was
           successfully defined in
           fileid fileno
```

The SAMPNSS EXEC first issues the DEFSYS command to define a skeleton system data file for CMS and for CMSXA. These messages tell you the DEFSYS commands were successful.

The names shown in the SAMPNSS command are the default names for the System/370 (CMS) and for the 370-XA (CMSXA) systems. Of course, you may name these systems whatever you wish; but if you change the names you must either edit the SAMPNSS EXEC to use your names or define your systems manually.

Note: The names are positional on the command (the System/370 name followed by the 370-XA name).

```
OWNERID  FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMS      NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The SAMPNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

3. Issue the QUERY NSS ALL MAP command to make sure CMS is defined properly. Check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Set your machine mode to 370, load CMS (IPL 190), and save the CMS system:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

If your machine is already in 370 mode, you will not get a response.

```
ipl 190 clear parm savesys cms ■
```

Load 190 with the option to save the system under the name CMS.

```
HCPNSD440I Named Saved System (NSS) CMS was
            successfully saved in fileid fileno
DMSWSP327I The installation saved segment could not be loaded
```

Disregard the message about not loading the saved segment; CMS has been successfully saved as a named saved system.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

5. Set your machine mode to XA, load CMS (IPL 190), and save the CMSXA system:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

If your machine is already in 370-XA mode, you do not get a response.

```
ipl 190 clear parm savesys cmsxa ■
```

Load 190 with the option to save the system under the name CMSXA.

```
HCPNSD440I Named Saved System (NSS) CMSXA was
           successfully saved in fileid fileno
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

Disregard the message about not loading the saved segment; CMSXA has been successfully saved as a named saved system.

```
■
```

Press **ENTER** to initialize CMS.

```
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the CMS (or CMSXA) named saved system. The example below shows loading CMS in System/370 mode:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

If your machine is already in 370 mode, you will not get a response.

```
define storage 2m ■
STORAGE = 2M
Storage cleared - system reset
```

Defining storage causes a system reset.

```
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYM
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

After receiving the CMS version identification, press **ENTER** to complete CMS or CMSXA initialization.

7. Check the CMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Check whether the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Redefine your storage before you continue with the following steps:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 28. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments

If you plan to use the CMS/DOS environment or the VSAM product, you must install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS saved segments.

If you do not want to install these saved segments, go to "Step 29. Create the CMSINST and HELP Saved Segments" on page 204.

In this step, you will:

- Install CMSDOS and CMSBAM in separate saved segments
- Create a CMSVSAM segment and install the VSAM product tape
- Create a CMSAMS segment and install Access Method Services.

This step takes approximately 1 hour.

Warning: The CMSDOS and CMSBAM segments must be installed before you install CMSVSAM and CMSAMS.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. IPL 190. This is the CMS system disk.

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
```

VM/XA CMS 5.6 *mm/dd/yy hh:mm* ** *DO NOT press ENTER!* **	Your own installation header appears if you have changed it.
access (noprof ■ Ready; T=*n.nn/n.nn hh:mm:ss*	This command suppresses execution of MAINT's PROFILE EXEC.
access 193 e ■ **access 5E5 b ■** Ready; T=*n.nn/n.nn hh:mm:ss*	Access the disks with installation and service EXECs, and the DOSGEN EXEC.
set emsg on ■ Ready; T=*n.nn/n.nn hh:mm:ss*	You want to see any and all error messages during execution of the installation EXEC.

2. Define a segment into which the DOSGEN EXEC can load CMSDOS:

defseg dosinst 900-90f sr ■ HCPNSD440I Saved segment DOSINST was successfully defined in fileid *fileno* Ready; T=*n.nn/n.nn hh:mm:ss*	This command defines a 1M segment for DOSINST starting at 900000. You may place this segment anywhere below the segment spaces defined for CMSDOS, CMSBAM, CMSVSAM, and CMSAMS.

3. Invoke the DOSGEN EXEC with a load address and the name DOSINST. The load address used for DOSINST in this example is 900000.

```
dosgen 900000 dosinst ■
HCPNSS440I Saved segment DOSINST was
            successfully saved in fileid
            fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWGN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the DOSINST segment. Error messages for the DOSGEN EXEC are listed on page 198.

4. IPL CMS:

```
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Define CMSDOS, CMSBAM, CMSAMS, and CMSVSAM:

```
set sysname cmsdos dosinst ■
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsdos ■
HCPNSD440I Saved segment CMSDOS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss DOSBAM   DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSDOS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsbam ■
HCPNSD440I Saved segment CMSBAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSBAM   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsams ■
HCPNSD440I Saved segment CMSAMS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSAMS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
sampnss cmsvsam ■
HCPNSD440I Saved segment CMSVSAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSVSAM  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Invoke the DOSGEN EXEC with a load address and the name CMSDOS. The load address set up by SAMPNSS is B00000 for the name CMSDOS.

```
access 193 e ■
Ready; T=n.nn/n.nn hh:mm:ss
```

```
dosgen b00000 cmsdos ■
HCPNSS440I Saved segment CMSDOS was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSGEN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the CMSDOS segment. Error messages for the DOSGEN EXEC are listed on page 198.

7. To save the load map, rename it and copy it to the alternate LOCAL minidisk (395):

```
access 395 d ■
DMSACC724I 395 replaces D(395)
Ready; T=n.nn/n.nn hh:mm:ss
copy load map a cmsdos segmap d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Error Messages from DOSGEN

If DOSGEN detects an error in the address that you specified:

```
DMSGEN095E INVALID ADDRESS
```

If DOSGEN cannot find a read/write accessed A-disk:

```
DMSGEN006E NO READ/WRITE A-DISK ACCESSED
```

If DOSGEN finds unresolved external references while loading the text files:

```
DMSGEN111E DOSGEN FAILED DUE TO LOAD ERRORS
```

If DOSGEN detects an error while assigning the storage key or saving the segment:

```
DMSGEN412S DOSGEN FAILED DUE TO SETKEY ERRORS

DMSGEN141S DOSGEN FAILED DUE TO SAVESYS ERRORS
```

8. Replace the name CMSBAM in the CMS SYSNAME table with any name that is **not** a name previously used as a segment name:

```
set sysname cmsbam sysname ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The SET SYSNAME command enters an alternate name for CMSBAM in the SYSNAME table. Choose a name that was **not** used previously as a segment name. This command does not actually change the saved segment's name, but keeps CP from finding and IPLing the segment at the wrong time.

9. Place your CMS virtual machine in a CMS/DOS environment, then invoke SAMGEN to load the CMSBAM segment. You must give the EXEC an address at which to load the CMSBAM segment; this address also must match the address in the skeleton CMSBAM segment that you defined beforehand.

```
set dos on ■
```

This command places your virtual machine in a CMS/DOS environment.

```
DMSSET400I SYSTEM systemname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
             at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
samgen ■
DMSSGN363R ENTER LOCATION WHERE CMSBAM
WILL BE LOADED AND SAVED:
b10000 ■
DMSSGN364I FETCHING CMSBAM...
DMSFET710I Phase DMSVBM entry point at location B100C0.
DMSSGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsbam ■
```

You may receive these informational messages. *sysname* is the name you chose in substep 8.

```
HCPNSS440I Saved segment CMSBAM was successfully
            saved in fileid fileno
DMSSGN365I SYSTEM CMSBAM SAVED
Ready; T=n.nn/n.nn hh:mm:ss
```

The messages indicate that the segment has been loaded and saved.

10. Set DOS off:

```
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

11. Define and format a minidisk for VSAM files.

    a. Map your USER DIRECT and examine the map to find a space for the new minidisk. Look for a gap of appropriate size and record the starting cylinder of that gap. The number of cylinders needed depends on the DASD type:

Device	Cylinders Needed
3350	15 cylinders
3375	22 cylinders
3380	14 cylinders
3390	14 cylinders

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
diskmap user direct ■
File USER DISKMAP A has been created.
Ready; T=n.nn/n.nn hh:mm:ss
xedit user diskmap ■
locate/GAP ■
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Refer to the directory map to find a gap. Repeat the **locate** command as many times as necessary. When you find a suitable gap, note the starting cylinder and DASD label the cylinders reside on and leave the file.

b. XEDIT your directory and add an entry for the new minidisk:

```
xedit user direct ■
locate/USER MAINT ■
locate/MDISK ■
input ■
MDISK vdevno devtype startcyl cyls label MW ALL ■
■
file ■
```

Locate the MAINT minidisk definitions.

vdevno, *devtype*, and *label* are the address, device type, and label of the DASD volume where the new minidisk is located. *startcyl* is the starting cylinder of the gap where the new minidisk is defined. *cyls* is the number of cylinders needed for the minidisk.

Press **ENTER** to return to the command line.

c. Map the directory again and examine the map to make sure that the new minidisk was created in the place you specified and that it does not overlap any existing minidisks:

```
diskmap user direct ■
```

d. Issue DIRECTXA to update the directory and bring it online:

```
directxa user ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ONLINE
Ready; T=n.nn/n.nn hh:mm:ss
```

e. Link to the new minidisk:

```
link * vdevno vdevno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

f. Format the new minidisk:

```
format vdevno v (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk V(vdevno).
            Do you wish to continue?
            Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
label ■
DMSFOR733I Formatting disk V
DMSFOR732I nn cylinders formatted on V(vdevno)
Ready; T=n.nn/n.nn hh:mm:ss
```

g. Access the new minidisk as A:

```
access vdevno a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

12. Set DOS on:

```
set dos on ■
DMSSET400I SYSTEM systemname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
             at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
```

You may receive these informational messages. *sysname* is the name you chose in substep 8 on page 199.

13. Mount the VSAM product tape on a tape drive attached to MAINT at virtual address 181. (If you do not have the proper class authority to attach a tape, the system operator may have to do it for you.)
14. Invoke VSAMGEN EXEC:

```
vsamgen ■
SELECT ONE OF THE FOLLOWING FUNCTIONS BY ENTERING THE NUMBER:

 1. INSTALL AMS            (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 2. INSTALL VSAM           (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 3. INSTALL VSAM AND AMS   (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 4. BUILD  AMS             (BUILD DOSLIB, CREATE SEGMENT)
 5. BUILD  VSAM            (BUILD DOSLIB, CREATE SEGMENT)
 6. BUILD  VSAM AND AMS    (BUILD DOSLIB, CREATE SEGMENT)
 7. RESTART AMS            (CREATE SEGMENT)
 8. RESTART VSAM           (CREATE SEGMENT)
 9. RESTART VSAM AND AMS   (CREATE SEGMENT)
10. QUIT                   (EXIT VSAMGEN EXECUTION)

      ENTER RESPONSE...
```

Choosing option 3 tells the EXEC to create both CMSVSAM and CMSAMS segments as new segments.

If this is the initial installation of VSAM, select function **1**, **2**, or **3**.

If the text files have already been created from the VSAM product tape and are currently on the accessed A-disk, select function **4**, **5**, or **6**.

If the DOSLIBs currently reside on an accessed disk, select function **7**, **8**, or **9**. In this case, it is not necessary to have the text files available.

```
3 ■

DMSVGN365R ONE OR MORE OF THE TEXT FILES LISTED IN THE BOTH EXEC
            ARE MISSING.  THE VSAM PP PID TAPE SHOULD BE ON TAPE DRIVE 181
            TO RESTORE THE FILES. ENTER:
            'GO' IF TAPE DRIVE IS READY TO LOAD FILES,
            'QUIT' TO STOP GENERATION PROCESS.
go ■
```

Messages from VSAMGEN

While VSAMGEN is processing, you may receive error and information messages. These messages are self-explanatory. Messages labeled 2101I are information messages from the linkage editor and may be ignored.

Respond to the EXEC messages as they appear:

```
DMSVGN362I LINK-EDITING CMSVSAM ...
DMSVGN363I CMSVSAM DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSVSAM SHARED WILL BE
         LOADED AND SAVED:
ba0000 ■

DMSVGN363R ENTER LOCATION WHERE CMSVSAM NONSHARED WILL BE
         LOADED AND SAVED:
a30000 ■

DMSVGN364I FETCHING CMSVSAM ...
DMSFET710I PHASE DMSVVS ENTRY POINT AT LOCATION nnnnnn
DSMVGN371R CMSVSAM IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsvsam ■

DMSVGN365I SYSTEM CMSVSAM SAVED.

DMSVGN368R ERASE CMSVSAM DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■

DMSVGN362I LINK-EDITING CMSAMS ...
DMSVGN363I CMSAMS DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSAMS SHARED WILL BE
         LOADED AND SAVED:
```

This name is the default name with which the segment was defined.

```
b40000 ■                                        This is the recommended location.
DMSVGN363R ENTER LOCATION WHERE CMSAMS NONSHARED WILL BE
         LOADED AND SAVED:
a00000 ■
DMSVGN364I FETCHING CMSAMS ...
DMSFET710I PHASE DMSVAS ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAN ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAX ENTRY POINT AT LOCATION nnnnnn
DMSVGN371R CMSAMS IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:

cmsams ■                                        This name is the default name used for defining
                                                the segment.
DMSVGN365I SYSTEM CMSAMS SAVED.

DMSVGN368R ERASE CMSAMS DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
Ready; T=n.nn/n.nn hh:mm:ss
```

15. You may now purge the DOSINST named saved segment:

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss

purge nss fileno ■                              fileno is the file identifier for DOSINST.
No files purged
0001 file pending purged
Ready; T=n.nn/n.nn hh:mm:ss
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 29. Create the CMSINST and HELP Saved Segments

In this step, you will:

- Define a segment to contain EXECs and System Product Editor macro instructions
- Use the DCSSGEN command to load and save the CMSINST segment
- Save the HELP file directory in a saved segment.

Notes:

1. You must have the NAMESAVE HELP statement in your user directory in order to save the HELP file directory information in a saved segment.

   See "Step 21. Update the User Directory" on page 71.

2. If the HELP segment is defined as a member of a segment space, you must have a NAMESAVE statement (for example, NAMESAVE INSTHELP) in the directory entry for the user who invokes SAVEFD. SAVEFD specifies the name of the segment space.

This step takes approximately 20 minutes.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. Log on as MAINT (unless you are continuing from the previous step).
2. Check your virtual storage. If it is less than 16M, issue the following:

```
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 0016M
```

3. IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
```

Press **ENTER** to complete the CMS initialization.

```
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Use the SAMPNSS EXEC to define a segment for CMSINST:

   Note: If you have increased the number of entries in the CMS load list, be sure that the DEFSEG entry for CMSINST in the SAMPNSS EXEC has enough pages defined to accommodate the increased size.

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
           defined in fileid fileno
```

```
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The EXEC issues a QUERY command that displays information about the segments defined for named saved systems.

5. Define the HELP segment:

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
           defined in fileid fileno
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Save both segments:

```
saveseg cmsinst ■
HCPNSS440I Saved segment CMSINST was
           successfully saved in fileid
           fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg help ■
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Each member saved segment defined by SAMPNSS must be saved separately.

7. Redefine each segment to create the skeleton segments:

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
           defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
           defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If you do not want to use the sample load list, CMSINST EXECLIST, for the DCSSGEN command, use the System Product Editor (XEDIT) to create your own load list:

```
xedit instlist file a ■
input ■
* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S ■
■
file ■
```

This is a sample. Enter the load list entries you need.

Only one comment line per list is allowed; it must be the first line. For more information on creating this list, see "DCSSGEN Command" on page 580.

The remainder of the lines contain entries for frequently used EXECs and Editor macros; multiple users can share the same executing copy of these EXECs and macros.

Press **ENTER** to return to the command line.

9. The load list must have a fixed record length in order for you to use the DCSSGEN command. If your load list has variable length records, change it to a fixed length file by issuing:

```
copyfile fn ft fm = = = (recfm f lrecl 80 replace ■
```

10. Invoke the DCSSGEN command:

```
access 193 e ■
dcssgen fn ft fm cmsinst ■
```

fn ft fm is the file ID of the file that contains the list of EXECs and Editor macros to be loaded into the segment, either CMSINST EXECLIST *fm* or your own file (for example, INSTLIST FILE A).

CMSINST is the name of the segment you want to create. It is also the default name of the installation saved segment in the DMSNGP profile. If you changed the name in the DMSNGP profile, or specified a different name in response to the installation questions (see page 152), use that name instead. If you do not specify a segment name, the DCSSGEN command defaults to CMSINST.

```
HCPNSS440I Saved segment CMSINST was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

The DCSSGEN command creates a load map file, CMSINST DCSSMAP A. For the sample load list created in substep 8 on page 206, the load map would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
    15:41:59 10/22/85 copy of RDRLIST  EXEC     S loaded as MAIL     EXEC
    EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
    15:41:58 10/22/85 copy of FILELIST EXEC     S loaded as FILELIST EXEC
    EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
     7:30:18 11/26/85 copy of SYSPROF  EXEC     S loaded as SYSPROF  EXEC
    EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
     8:47:55 12/18/84 copy of PARSE    XEDIT    S loaded as PARSE    XEDIT
    EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
    15:40:32 10/22/85 copy of DISCARD  EXEC     S loaded as DISCARD  EXEC
    EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
    11:03:53 10/24/85 copy of NOTE     EXEC     S loaded as NOTE     EXEC
    EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
    15:41:55 10/22/85 copy of PROFNOTE XEDIT    S loaded as PROFNOTE XEDIT
    EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
    15:41:05 10/22/85 copy of ALL      XEDIT    S loaded as ALL      XEDIT
    EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
    *** End of Source List ***
    CMSINST built at 15:56:34 on 12/02/85
```

Note: When you built your CMS nucleus, if you indicated in the DMSNGP file (USEINST = YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs CMS. If you previously indicated that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild CMS. (To find the latest version of the DMSNGP file, issue SETUP to access all the minidisks where it might be; then issue FILELIST DMSNGP ASSEMBLE *. All the copies of DMSNGP ASSEMBLE will be listed, with the date when each one was last modified.)

Messages from DCSSGEN

While DCSSGEN is processing, you may receive error or warning messages that indicate specific conditions. If errors were encountered, after processing is complete you receive the following prompt:

```
DMSEXG298R An error has been detected while building the DCSS.
           Do you still want the DCSS saved?
           Enter 1 (YES) or 0 (NO).
```

Enter **1** to disregard the errors and save the segment, or enter **0** to not save the segment. If you do not save the segment, you receive the message:

DMSEXG288I *segname* not saved

If DCSSGEN encounters an error while saving the segment, you receive the message:

DMSEXG288E *dcssname* not saved

If your virtual machine is not large enough to contain the segment (you need 16M), you receive the message:

```
DMSEXG284E The DCSS is not completely inside the virtual machine
```

To correct this situation, increase the size of your virtual machine, re-IPL CMS, and reissue the DCSSGEN command.

11. Define your virtual storage **less** than the address at which the HELP segment is to be loaded. For example, if the HELP segment is defined at X'C50000' (the default address in the SAMPNSS EXEC) define your storage as 12M.

```
define storage 12m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 12M
```

12. Re-IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

13. Issue the following commands to initialize and save the segment:

    (For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*.)

```
savefd init vaddr label help ■
Ready; T=n.nn/n.nn hh:mm:ss
savefd save vaddr label help ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is 19D for mixed-case American English, 19C for uppercase American English.

label is the CMS label assigned to the disk. (The labels in the sample directory are MNT19D and MNT19C.)

14. Verify that the CMSINST and CMSHELP segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss INSTHELP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss CMSINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss HELP        NSS      MAINT
```

The system data file should now have class A (rather than S) and have one user (MAINT).

15. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 30. Back Up the Named Saved System Files

In this step, you will back up all the named saved systems, including CMS, on tape.

This step takes approximately 10 minutes.

1. Mount a scratch tape with ring on an available 3420, 3422, 3430, or 3480 tape drive, or insert a cartridge with the tab protector set to off in an available 3480 tape unit. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT.
   - The tape must be suitable for a 3420, 3422, 3430, or 3480 tape unit.
2. Issue the SPTAPE command to dump the CMS system data file to tape:

   ```
   sptape dump vdevno nss all run ■
   ```

   Substitute the virtual device number of the tape drive for the value *vdevno*. The operand **run** specifies that the SPTAPE rewinds and unloads the tape after the operation.

   ```
   DUMPING vdevno NSS *NSS      nnnn A
   ⋮
   SPTAPE DUMP FUNCTION ON DRIVE vdevno
      COMPLETE
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   The messages from SPTAPE tell you that the file is being dumped to tape. The CMS ready message may occur between the messages.

3. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE command to restore the CMS system data file. For more information about the SPTAPE command, see *VM/XA SP Real System Operation.*

To restore your CMS named saved system:

1. Log on as MAINT.
2. Mount the backup tape on a tape drive.
3. Issue:

   ```
   sptape load vdevno nss all run ■
   LOADING vdevno NSS *NSS nnnn A    NOW nnnn
   ⋮
   LOADING vdevno NSS *NSS nnnn A    NOW nnnn
   SPTAPE LOAD FUNCTION ON DRIVE vdevno COMPLETE
   ```

4. IPL CMS:

   ```
   ipl cms ■
   VM/XA CMS 5.6 mm/dd/yy hh:mm
   ■
   SYNONYM SYN
   CP TERM MODE VM
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

Step 31. Store a Backup Copy of CP on Tape

In this step, you will use DDRXA to store a backup copy of CP on tape. The time to perform this step varies based on the device and contention for the device's channel. It may take up to 30 minutes.

1. Mount a scratch tape (with ring) on a tape drive. Follow machine operations manuals to mount the tape.
2. Attach the tape drive to MAINT at virtual device number 0181:

```
attach vdevno * 181 ■
TAPE     0181 ATTACHED
Ready; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the virtual device (*vdevno*) to MAINT's virtual machine at virtual device number 0181.

3. Load the DDRXA utility to tape. If your tape drive can operate at a density of 6250 BPI, enter the commands under 3a. Otherwise, enter the commands under 3b.
   a. If your tape drive can operate at a density of 6250 BPI, enter these commands:

```
tape modeset (tapl den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Set the tape density to 6250 BPI. If you receive message DMSPAR622E, reissue the TAPE MODESET command.

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl (den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

Go to substep 4.

   b. If your tape drive cannot operate at a density of 6250 BPI, enter these commands:

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

4. Change MAINT's virtual processor configuration to 370-XA architecture:

```
set machine xa ■
System reset
System = XA
```

Setting the virtual machine to 370-XA architecture causes a reset as if you entered SYSTEM CLEAR. If your machine is already in 370-XA mode, you will not get a response.

```
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Rewind the scratch tape on virtual device number 0181:

```
rewind 181 ■
Rewind complete
```

6. Load (IPL) the tape and answer the prompts from DDRXA:

```
ipl 181 clear ■
```

Wait a few moments for DDRXA to prompt you. If a prompt does not appear, press the **ENTER** key.

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:
```

```
sysprint cons ■
ENTER:
```

This first control statement tells DDRXA that you want program messages sent to your console.

```
input vdevno devtype xasres ■
ENTER:
```

The second control statement is the input control statement. *vdevno* is the virtual address of the system residence volume. (The virtual address in the sample HCPSYS ASSEMBLE file and USER DIRECT is 123.) *devtype* is the DASD type of this volume.

```
output 181 devtype (mode 6250 compact ■
ENTER:
```

This control statement specifies the device to which you are dumping the system.

If your tape drive is not operating at 6250 BPI density, omit the **mode 6250 compact** option on this command.

```
dump cpvol ■
```

The statement DUMP CPVOL dumps cylinder 0 and all PERM and DRCT DASD space from the system residence volume to the tape.

```
DUMPING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn    0000   nnnn
END OF DUMP
BYTES IN nnnnnnnnnn BYTES OUT nnnnnnnnnn
TRACKS NOT COMPACTED ON TAPE - nnnnnnnnnn
```

These are informational messages. GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

```
ENTER:
■
END OF JOB
```

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

You now have a loadable tape of the system. If you need to use this backup copy to restore CP, perform these steps **on a first-level system**:

1. Mount the backup tape on a tape drive.
2. Set your virtual machine to 370-XA architecture, define your virtual storage as 16 megabytes, and IPL CMSXA:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
define storage 16m ■

STORAGE = 16M
Storage cleared - system reset
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If your machine is already in 370-XA mode, you will not get a response.

3. Perform a hardware load (IPL) of the tape device.
4. Use DDRXA to restore the system to disk:

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:

sysprint cons ■
ENTER:

input vdevno devtype ■
ENTER:

output vdevno devtype xasres ■
ENTER:

restore all ■

RESTORING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO XASRES
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn     0000   nnnn
⋮
END OF RESTORE
BYTES RESTORED nnnnnnnnn

ENTER:
■
END OF JOB
```

This first control statement tells DDRXA that you want program messages sent to your console.

The second control statement is the input control statement. Identify the device number (*vdevno*) and tape device type (3420, 3422, 3430, or 3480) where the backup tape is mounted.

This control statement specifies the DASD device to which you are restoring the system (XASRES).

The RESTORE ALL statement tells DDRXA to restore the whole tape to the output device.

Informational messages: GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

Chapter 4. Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System

If you have a VM/SP or VM/SP HPO system, you can use it to install VM/XA System Product Release 2.1. Otherwise, you must use the Starter System. See Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" on page 13 or Chapter 3, "Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)" on page 113.

If your VM/SP or VM/SP HPO system is running as a second-level guest of VM/XA MA, VM/XA SF, or VM/XA SP, some of the messages will differ from the ones shown in this procedure.

Before You Begin

Before you begin to install VM/XA SP, be sure to read the *VM/XA SP Conversion Notebook*.

You may wish to do the following steps now. You can also do them in the order in which they appear in this chapter.

- "Step 16. Convert the DMKRIO and DMKSYS ASSEMBLE Files and Tailor HCPBOX ASSEMBLE and the Product Parameter File" on page 254
- "Step 19. Create the User Directory" on page 263.

Step 1. Make a Directory Entry for XAMAINT

In this step, you will make an entry for the VM/XA SP installation ID (XAMAINT) in your VM/SP or VM/SP HPO directory.

This step takes approximately 5 minutes.

In order to install VM/XA SP using your existing system, you must define the VM/XA SP installation ID (XAMAINT) in your VM/SP or VM/SP HPO directory.

1. Mount the first volume of the VM/XA SP product tape on a tape drive and attach the tape drive to your user ID as 181:

   ```
   attach vdevno * 181 ■
   ```

2. Get the file containing a sample directory entry for XAMAINT. (A sample XAMAINT directory entry for each type of DASD is printed in Appendix C, "VM/XA System Product Starter System Information," after the corresponding sample directory.) Issue:

   ```
   vmfplc2 load 33xx xamaint (eof 6 ■
   LOADING.....
   END-OF-FILE OR END-OF-TAPE
   END-OF-FILE OR END-OF-TAPE
   END-OF-FILE OR END-OF-TAPE
   END-OF-FILE OR END-OF-TAPE
   END-OF-FILE OR END-OF-TAPE
   33xx XAMAINT A1
   END-OF-FILE OR END-OF-TAPE
   READY; T=n.nn/n.nn hh:mm:ss
   ```

   33xx is 3350, 3375, 3380, 3383 (for 3380-K), 338E (for 3380-E4), 3390 (for 3390-1), or 3391 (for 3390-2).

3. XEDIT your existing directory to add the entry for XAMAINT:

   ```
   xedit filename direct ■
   bottom ■
   get 33xx xamaint ■
   set wrap on ■
   locate /XAMAINT ■
   ```

   filename is the filename of your directory.

   ```
   change /NOLOG/password ■
   file ■
   ```

   Change NOLOG to a password of your choice.

4. Issue the DIRECT command to bring the edited directory on line:

   ```
   direct filename ■
   EOJ DIRECTORY UPDATED AND ON LINE
   READY; T=n.nn/n.nn hh:mm:ss
   ```

Step 2. Restore the VM/XA System Product Starter System to Disk

In this step, you will restore the starter system to disk.

This step takes approximately 15 minutes, if you do not format any DASD. Formatting DASD volumes can take up to 30 minutes per volume.

Although you are not using the Starter System to build your new VM/XA SP system, load it and restore it to disk now. Restoring the Starter System initializes your system residence volume.

Warning: Do not build the VM/XA System Product system residence volume on the same volume as your current system residence volume.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, refer to the Program Directory that you received with the product tape for the most current information.

1. Ready the DASD you plan to use. XASRES should be formatted already. If it isn't formatted, format it now.

   Warning: The addresses of XASRES, XASERV (if used), XAP001 (if used), and XAP002 (if used) must match the addresses in the HCPRIO ASSEMBLE file. If the addresses do not match, you will not be able to IPL your new system in Step 23. You must either convert your existing DMKRIO ASSEMBLE file or tailor the sample HCPRIO ASSEMBLE file in Step 16.

2. Log on as XAMAINT:

   Warning: When you log on as XAMAINT, you must not have any packs labeled XASRES, XASERV, XAP001, or XAP002 online, **including the ones you intend to use during the installation process.** If you have any such packs, you will not be able to use the ones you need. If necessary, you can use the Device Support Facilities to relabel your packs. Issue REFORMAT UNIT(*vdevno*) VERIFY(*oldlabel*) VOLID(*newlabel*) when DSF prompts you with `ENTER INPUT/COMMAND`.

   ```
   logon xamaint ■
   ENTER PASSWORD
   password ■
   DMKLNM108E XAMAINT 124 not linked; volid XAP001 not mounted
   DMKLNM108E XAMAINT 125 not linked; volid XASERV not mounted
   DMKLNM108E XAMAINT 126 not linked; volid XAP002 not mounted
   ```

   You will see this message only if you are using 3350 DASD.

   ```
   DASD 390 linked R/O, R/W by MAINT, R/O by OPERATOR
   LOGON AT hh:mm:ss
   Device 190 does not exist
   ```

3. Enter:

   ```
   terminal mode vm ■
   spool console * start ■
   ```

4. IPL CMS:

```
define 390 190 ■
DASD 190 DEFINED
ipl 190 clear ■
DMSINS327I The installation DCSS could not be loaded
VM/SP RELn mm/dd/yy hh:mm
■
DMSACP1135 A(191) NOT ATTACHED
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Attach the volume that will become your VM/XA SP system residence volume (XASRES) to XAMAINT:

```
vary online vdevno ■
vdevno VARIED ONLINE
```

vdevno is the *virtual* device address of XASRES.

```
attach rdevno * ■
DASD vdevno ATTACH TO XAMAINT  vdevno
```

rdevno is the *real* device address of XASRES.

6. Attach a tape drive to XAMAINT as 181:

```
attach rdevno *  181 ■
TAPE rdevno ATTACH TO XAMAINT  181
```

rdevno is the real device address of the tape drive.

7. Mount the VM/XA SP Starter System tape on the tape drive.
8. Restore the VM/XA SP Starter System to disk:

```
tape fsf ■
READY; T=n.nn/n.nn hh:mm:ss
```

This command bypasses the first tape file, which contains the Device Support Facilities and the DDRXA program. **Do not** try to bypass this file by IPLing the tape, since DDRXA will not run in 370 mode.

```
ddr ■
ENTER:
sysprint cons ■
ENTER:
```

```
input 181 devtype ■
ENTER:
```

181 and *devtype* are the virtual address and the device type (3420, 3422, 3430, or 3480) of the tape drive where the starter system tape is mounted.

```
output vdevno 33xx scratch ■
ENTER
restore all ■
```

vdevno is the virtual address of XASRES. 33*xx* is 3350, 3375, 3380, or 3390.

```
DMKDDR725R DASD INPUT DEVICE WAS(IS) LARGER THAN OUTPUT DEVICE
DO YOU WISH TO CONTINUE? RESPOND YES OR NO:
yes ■
RESTORING XASRES
DATA DUMPED mm/dd/yy at hh.mm.ss GMT
   FROM XASRES RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS    OUTPUT CYLINDER EXTENTS
       START     STOP            START     STOP
       cccc      cccc            cccc      cccc

⋮
END OF RESTORE
BYTES RESTORED nnnnnnnnn
ENTER:
■
END OF JOB
Ready; T=n.nn/n.nn hh:mm:ss
```

Depending on the DASD model you are using, you may or may not see this prompt. If you do see it, respond **yes**. The starter system is designed to fit on all supported DASD models.

9. When the restore is complete, detach XASRES from XAMAINT and attach it to the system:

   Note: Detach the *virtual* address, but attach the *real* address.

```
detach vdevno ■
DASD vdevno DETACHED
Ready; T=n.nn/n.nn hh:mm:ss
attach rdevno system xasres ■
DASD rdevno ATTACH TO SYSTEM XASRES
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Log off to bring the Starter System CMS nucleus online.

Step 3. Attach the VM/XA System Product Release 2.1 Product Tape (Volume 1) and Load the First Three Files

In this step, you will:

- Mount Volume 1 of the VM/XA System Product product tape on a tape drive
- Attach the product tape drive to XAMAINT
- Load the first three files from the product tape.

This step takes approximately 5 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Log on as XAMAINT:

```
logon xamaint ■
ENTER PASSWORD
password ■
```

```
THE NLSAMENG SAVED SEGMENT COULD NOT BE FOUND;
  RETURN CODE 44 FROM SEGMENT
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
```

Ignore this message. There is nothing wrong.

You are now running on the VM/XA System Product version of CMS.

2. Mount Volume 1 of the VM/XA System Product product tape (in write-protect mode) on a tape drive. Follow the operation manual for the machine on which you mount the tape.
3. Attach the tape drive to XAMAINT at virtual device number 0181:

```
attach vdevno * 181 ■
TAPE vdevno ATTACH TO XAMAINT 181
READY; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the virtual device (*vdevno*) to XAMAINT's virtual machine at virtual device number 0181.

4. Check XAMAINT's disk accesses. Be sure XAMAINT's 191 minidisk is the A disk.

```
query search ■
MNT191   191 A    R/W
MNT190   190 S    R/O
READY; T=n.nn/n.nn hh:mm:ss
```

The QUERY SEARCH command tells you what disks you have accessed and the alphabetic order of those accesses. When searching for files, CMS uses the alphabetic order of accesses to create a search order. CMS searches the A minidisk, if accessed, then the B minidisk, if accessed, and so forth.

5. If XAMAINT's 191 minidisk is not A, access that minidisk as A:

```
access 191 a ■
191 REPLACES A (195)
READY; T=n.nn/n.nn hh:mm:ss
```

This message would appear if the QUERY SEARCH command had shown MNT195 195 A R/W.

6. For your own convenience, set **PF12** as a retrieve key. When you press the retrieve key, the last instruction you issued will be displayed. You need only press **ENTER** to enter it again.

```
set pf12 retrieve ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You may add this command to your PROFILE EXEC.

7. Use the VMFPLC2 LOAD command to load the first three files (the tape header, memos, and installation tools) from the product tape to your A-disk:

```
vmfplc2 load (eof 3 ■
 LOADING.....
 $TAPE$   HEADER   A1
 END-OF-FILE OR END-OF-TAPE
```

This command loads the first three tape files from the product tape. (A tape file is the set of files between two tape marks.) These are the only tape files you have to load—the ITASK EXEC automatically loads the rest of the tape. You will receive a message from VMFPLC2 telling you which files it loaded.

```
 SUPSP21  MEMO     A1
 56643089 SERVICE  A1
 END-OF-FILE OR END-OF-TAPE
```

The VMSUP memo is loaded to the MAINT 191 minidisk. Read this memo before going on.

After you perform Step 7, you may wish to copy SUPSP21 MEMO to the CP BASE disk (194 or 394) to free space on MAINT 191.

```
 56643089 $PPF     A1
 RPWLIST  DATA     A1
 $MSG4I$  EXCAMENG A1
 $$RDPW$$ EXEC     A1
 $MSG4I$  EXEC     A1
 SETUP    EXEC     A1
 SPLOAD   EXEC     A1
 UTILITY  EXEC     A1
 ITASK    EXEC     A2
 DIRECTXA MODULE   A2
 DVM      PROFILE  A1
 SPLOAD   PROFILE  A1
 END-OF-FILE OR END-OF-TAPE
READY; T=n.nn/n.nn hh:mm:ss
```

The tools in the third file are installation EXECs. You may erase them **after you finish the installation**, except for these files:

- DIRECTXA MODULE, which is erased during the installation procedure
- SETUP EXEC, which **must** remain on the XAMAINT 191 minidisk.

The DIRECTXA MODULE is loaded to XAMAINT 191 now so that you can bring your directory online in Step 4. It is erased after it is used, then reloaded to XAMAINT 190 in Step 7. If you need to reissue ITASK ALLOCATE, you have to rewind the tape and issue VMFPLC2 LOAD DIRECTXA MODULE A (EOT to reload the DIRECTXA MODULE from Volume 1 of the product tape.

Step 4. Invoke the ITASK EXEC

In this step, you will use the ITASK EXEC to:

- Provide a password for user IDs MAINT and OPERATOR
- Load the sample directory and system definition files
- Format the XASERV, XAP001, and XAP002 volumes.

The time for this step depends upon the devices involved. It usually takes at least 30 minutes for each pack you format.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

Note: A description of the ITASK EXEC and its operands can be found in "ITASK EXEC" on page 611.

1. Make your DASD volumes available for use; issue the following command for each volume except XASRES:

   `vary online` *vdevno* ■

   vdevno is the **first-level virtual** address of the volume; this address functions as a "real" address at second level.

2. Issue ITASK ALLOCATE to supply a password for MAINT and OPERATOR and to format the XASERV XAP001, and XAP002. volumes.

   ITASK EXEC issues the DIRECTXA command after prompting you for passwords for MAINT and OPERATOR. The user ID running the ITASK EXEC should **not** have a link to the SYSRES volume for the first-level system.

   Note: The example in this manual uses a 3380 DASD for the minidisks. If you use a 3350, 3375, 3380-E4, 3380-K, or 3390, your space allocations and starting address for CMS will differ.

   `itask allocate` ■

   ```
   REWIND COMPLETE
   DMSACC724I 191 REPLACES A (191)
   REWIND COMPLETE
   DMSWTK409I LOADING 33xx * TO 191 ATTACHED TO XAMAINT
   ```

   These are informational messages to tell you how the operation is progressing. *xx* is 50, 75, 80, 8E, 83, 90, or 91, according to your DASD type.

   Note: If ITASK exits with return code 1040, the 123 full-pack minidisk is not attached. The virtual address of XASRES is 123. You may have to reattach XASRES to your system.

3. When prompted, enter a new password to be used as both the MAINT and OPERATOR passwords for your new VM/XA SP system:

```
ENTER ONE PASSWORD FOR MAINT AND OPERATOR:
password ■
```

You must supply a single password to be used for both the MAINT and OPERATOR user IDs. The ITASK EXEC will XEDIT the user directory and replace NOLOG with the logon password that you provide here.

When selecting a password, refer to the RPWLIST DATA file shown in Appendix F, "Restricted Logon Passwords" on page 859. This file contains all of the restricted passwords for your VM/XA SP2.1 system. You cannot choose a password for MAINT and OPERATOR that exists in this file.

```
ENTER PASSWORD AGAIN TO VERIFY:
password ■
```

Reenter the password that you have designated for the MAINT and OPERATOR user IDs.

YOU NEED TO REMEMBER THIS PASSWORD FOR USE IN LATER STEPS. You can write it down on the worksheet—see Table 1 on page 7.

```
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION
   PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED
DMSWSL970I FORMATTING XAMAINT 193 MINIDISK
DMSWSL409I LOADING SYSGEN TOOLS TO THE 193 DISK ATTACHED TO XAMAINT
```

These first messages signify that the user directory has been created. Then ITASK loads the system generation tools needed next.

```
DMSWSL970I FORMATTING XAMAINT 295 MINIDISK
DMSWSL409I LOADING HCPSYSxx * TO THE 295 DISK
             ATTACHED TO XAMAINT
DMSWSL409I LOADING HCPRIOXA * TO THE 295 DISK
             ATTACHED TO XAMAINT
DMSWSL409I LOADING HCPBOX * TO THE 295 DISK
             ATTACHED TO XAMAINT
DMSWSL970I FORMATTING XAMAINT 395 MINIDISK
DMSWSL409I LOADING CMSLOC SAMPLES TO THE 395 DISK
             ATTACHED TO XAMAINT
DMSACC724I 191 REPLACES A (191)
```

These are messages from ITASK as it loads files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

```
DMSWTK965I YOU MAY WISH TO TAILOR THE FOLLOWING
           FILES BEFORE NUCLEI GENERATION:

           HCPRIO ASSEMBLE
           HCPBOX ASSEMBLE
           HCPSYS ASSEMBLE
           DMSNGP ASSEMBLE
           USER  DIRECT
```

Instead of tailoring the sample files, you will probably want to convert the system definition files from your current system. To do this, consult *VM/XA SP Conversion Notebook*. You will have a chance to do this before generating the nucleus. You do not need to do it now.

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XASERV VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

vdevno or skip ■

Enter the virtual address, *vdevno*, of the XASERV volume. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you do not want to use XASERV, or if you have already formatted it, enter **skip**.

```
DASD vdevno ATTACHED TO XAMAINT 301
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 301

DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 301
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
CYLINDERS nnnn THROUGH nnnn FORMATTED
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XASERV'

HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED XAMAINT 0301
DASD vdevno ATTACHED TO SYSTEM XASERV

DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP001 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

This message signifies that CPFMTXA has completed formatting the XASERV volume.

The HCPFAM384I message signals the completion of this sequence.

vdevno or skip ■

```
DASD vdevno ATTACHED TO MAINT 302
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 302
```

Enter the **virtual** address, *vdevno*, to allocate the XAP001 volume. If you are using 3350 or 3375 DASD, you must have XAP001. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER**.

If you do not want to use XAP001, or if you have already formatted it, enter **skip**.

Note: If you do not have XAP001, you will receive this message:

```
HCPLND108E MAINT 0124 NOT LINKED
```

when you log on as MAINT. As long as you are using 3380 DASD, you can ignore this message.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 302
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP001'
```

This message signifies that CPFMTXA has completed formatting the XAP001 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED XAMAINT 0302
DASD vdevno ATTACHED TO SYSTEM XAP001
```

The HCPFAM384I message signals the completion of this sequence.

3350 DASD Only

```
DMSWTK981I WHAT IS THE REAL ADDRESS OF YOUR XAP002 VOLUME?
DMSWTK982I TYPE: REAL ADDRESS OR SKIP
```

```
vdevno or skip ■
DASD vdevno ATTACHED TO MAINT 303
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 0000-nnnn ON DISK 303
```

Enter the virtual address, *vdevno*, to allocate the XAP002 volume. The 3350 starter system requires XAP002. Refer to your worksheet (Table 1 on page 7) for the label and address of this system volume. **Check that you typed the right address before you press ENTER.**

If you have already formatted XAP002, enter **skip**.

```
DO YOU WANT TO CONTINUE ? (YES | NO)
FORMAT IN PROGRESS ON DISK 303
CYLINDERS 0 THROUGH 49 FORMATTED
⋮
```

You need not answer; the EXEC answers YES.

```
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
CURRENT ALLOCATION
TYPE CYLINDERS
.................
PERM 0000-nnnn
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XAP002'
```

This message signifies that CPFMTXA has completed formatting the XAP002 volume.

```
HCPFAM384I CPFMTXA  COMPLETE
DASD vdevno DETACHED XAMAINT 0303
DASD vdevno ATTACHED TO SYSTEM XAP002
```

The HCPFAM384I message signals the completion of this sequence.

End of 3350 DASD Only

```
READY; T=n.nn/n.nn hh:mm:ss
```

This message tells you that the ITASK EXEC has completed the sequence for each volume needed for installing VM/XA System Product Release 2.1.

Step 5. Format the Remaining Base CP Minidisks

In this step, you will use the ITASK EXEC to format the remaining minidisks that are defined in the CP directory, but have not yet been formatted or loaded in a previous step.

The time to perform this step may vary, but a good estimate is 45 minutes.

1. Issue SET ECMODE ON to insure that all the minidisks defined will be recognized:

```
set ecmode on ■
CP ENTERED; DISABLED WAIT PSW '00020000 00000000'
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
```

2. Use the ITASK EXEC with the BASEIDS parameter to format, load, and add (as appropriate) PROFILE EXECs to the remaining minidisks in the base CP directory. If you issue ITASK BASEIDS from a user ID other than XAMAINT, it will format that user ID's minidisks instead of XAMAINT's.

   Note: The block sizes of the 190 and 490 minidisks have increased from 1KB to 4KB for minidisk caching. The Starter System defines the 190 and 191 minidisks (formatting the 191 disk with a 1KB block size), while the ITASK BASEIDS step formats all other minidisks with a 2KB block size.

   If you choose to format a minidisk with a block size of 4KB, the number of cylinders for that minidisk may have to be increased, as is the case with the HELP disks.

```
itask baseids ■
DMSWTK1310I DO YOU WANT TO FORMAT MINIDISKS:
             AUTOLOG1 (191), CMSBATCH (195), DISKACNT (191),
             XAMAINT 36E (PVM), EREP (191), AND OPERATNS (191)?
DMSWTK8005R TYPE: 1 (YES) OR 0 (NO).
0 ■
```

You cannot format these minidisks now. If you answer **yes**, your **existing system's** minidisks will be formatted. You must format your new system's minidisks after the new system is brought online.

```
DMSWTK986I THE FOLLOWING MINIDISKS
           DEFINED IN THE BASE CP DIRECTORY
           WILL BE FORMATTED:

           XAMAINT  292     XAMAINT  49C
           XAMAINT  194     XAMAINT  49D
           XAMAINT  19C     XAMAINT  501
           XAMAINT  19D     XAMAINT  591
           XAMAINT  19E     XAMAINT  592
           XAMAINT  201     XAMAINT  593
           XAMAINT  291     XAMAINT  594
           XAMAINT  293     XAMAINT  595
```

```
              XAMAINT  294     XAMAINT  596
              XAMAINT  391     XAMAINT  59E
              XAMAINT  392     XAMAINT  5E5
              XAMAINT  393     XAMAINT  691
              XAMAINT  394     XAMAINT  692
              XAMAINT  423     XAMAINT  791
              XAMAINT  490     XAMAINT  892
              XAMAINT  491     XAMAINT  895
              XAMAINT  492     XAMAINT  896
              XAMAINT  495     XAMAINT  89E

    DMSWTK970I FORMATTING XAMAINT 292 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 194 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 19C MINIDISK
    DMSWTK970I FORMATTING XAMAINT 19D MINIDISK
    DMSWTK970I FORMATTING XAMAINT 19E MINIDISK
    DMSWTK970I FORMATTING XAMAINT 201 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 291 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 293 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 294 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 391 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 392 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 393 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 394 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 423 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 490 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 491 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 492 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 495 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 49C MINIDISK
    DMSWTK970I FORMATTING XAMAINT 49D MINIDISK
    DMSWTK970I FORMATTING XAMAINT 501 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 591 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 592 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 593 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 594 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 595 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 596 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 59E MINIDISK
    DMSWTK970I FORMATTING XAMAINT 5E5 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 691 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 692 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 791 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 892 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 895 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 896 MINIDISK
    DMSWTK970I FORMATTING XAMAINT 89E MINIDISK

    DASD 333 DETACHED

    READY; T=n.nn/n.nn hh:mm:ss
```

The ready message indicates that ITASK has completed its work successfully.

3. If any of the minidisks listed in message DMSWTK968I were not formatted, check your work packs to make sure that they are labeled XASERV, XAP001, and XAP002, and that they are attached to the system. Then reissue ITASK BASEIDS.
4. Continue with the next step.

Step 6. ITASK Loads Files from the Product Tape (Volume 1)

In this step, you will issue the ITASK EXEC to load files from Volume 1 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Enter the ITASK command with the LOAD ALL1 operands to load the CP and dump viewing facility object code and service files. If a problem should occur during this step, or if for some reason you want to return to this step and load product tape files, see "ITASK EXEC" on page 611 for how to load the files you need.

```
itask load all1 ■
DMSWSL409I LOADING SYSTEM SAMPLES TO
     THE 191 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CP OBJECT TO
     THE 194 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CP PUTAPPLY TO
     THE 292 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CP PUTDELTA TO
     THE 294 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CP CORAPPLY TO
     THE 292 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CP CORDELTA TO
     THE 294 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING DUMPVIEW OBJECT TO
     THE 193 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING DUMPVIEW PUTAPPLY TO
     THE 392 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING DUMPVIEW PUTDELTA TO
     THE 293 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING DUMPVIEW CORAPPLY TO
     THE 392 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING DUMPVIEW CORDELTA TO
     THE 293 DISK ATTACHED TO XAMAINT
READY; T=n.nn/n.nn hh:mm:ss
```

These are messages from ITASK as it loads more files from the product tape. The time the ITASK EXEC needs to execute varies from system to system.

Step 7. ITASK Loads Files from the Product Tape (Volume 2)

In this step, you will issue the ITASK EXEC to load files from Volume 2 of the product tape.

This step takes approximately 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all ot them. We recommend that you do not change any addresses.

1. Mount Volume 2 of the product tape.
2. Issue ITASK LOAD CMS to load the CMS object code, source code, and service files; and the IOCP files:

```
itask load cms ■
DMSWSL409I LOADING CMS BASE TO
     THE 193 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING IOCP FILES TO
     THE 193 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CMS PUTAPPLY TO
     THE 392 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CMS PUTDELTA TO
     THE 293 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CMS CORAPPLY TO
     THE 392 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CMS CORDELTA TO
     THE 293 DISK ATTACHED TO XAMAINT
DMSWSL409I LOADING CMS SYSTEM TO
     THE 190 DISK ATTACHED TO XAMAINT
```

3. The ITASK EXEC invokes the SETUP EXEC to establish the minidisk access order, then the ASMGEND EXEC to build the system F assembler and create the associated auxiliary directory:

```
DMSACC724I 191 REPLACES A (191)
⋮
DMSACC724I 190 REPLACES A (191)
DMSACC725I 190 ALSO = S DISK
ASSEMBLE XF GEND PROC
ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED.
■
```

Press **ENTER** to place the modules on the S-disk, the system default.

Enter another disk-mode letter if you prefer the modules to be placed on another minidisk.

```
ASSEMBLE XF GEND COMPLETE

DMSACC724I 191 REPLACES A (190)
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

a. At this time, you may choose to make a copy of the documentation file (SUPSP21 MEMO) to the CP BASE disk as suggested in Step 3, substep 7 on page 221.

b. You may choose to load HELP files, CP, CMS, and dump viewing facility source files, and GCS object files at this time. The following is an alternative order for loading these files that will minimize tape mounts:

   1) Load HELP files from Volume 2. (See Step 24 on page 270.)

   2) Load GCS object files from Volume 2. (See Step 28, substep 3 on page 276.)

   3) Load CP, CMS, and dump viewing facility source files from Volume 3. (See Step 25 on page 271.)

Step 8. Establish the Service Tools Build Disk

In this step, you will copy the latest level of the service tools to the service tools build disk.

This step takes approximately 10 to 15 minutes.

To be sure that you are using the most recent service EXECs, you must have the latest level of service on a service tools build disk. This disk, 5E5, is defined in the sample directories. It is the only disk in the TASK string in the product parameter file.

1. IPL 190:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order and create a work EXEC to help you copy the service tools:

```
setup ■
Ready; T=n.nn/n.nn hh:mm:ss
DMSACC724I vdevno replaces fm (vdevno
⋮
vmfsetup 56643089 cms ■
DMSACC724I vdevno replaces fm (vdevno
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
xedit lvlses exec fm ■
input ■
```

fm is the filemode of the task disk (5E5).

```
&TRACE OFF
&FT = &LEFT OF &2 2
&FT = &CONCAT OF &FT *
EXEC FILELIST &1 &FT *
&EXIT 0
```

Enter these lines exactly as they appear.

```
■
■
file ■
```

Press **ENTER** twice to return to the command line.

3. Look for the latest version of the CPYSVSES EXEC or CPYSVSES EXC*nnnnn*. CPYSVSES contains a list of all the service EXECs.

```
filelist cpysvses * * ■
```

Examine each version of CPYSVSES. CPYSVSES has an update history at the top. The latest level is one with the most updates. Use the update history, **not the date and time stamp**, to determine which is the most recent version.

4. Copy the latest level of CPYSVSES to the task disk (5E5) with a filetype of EXEC:

```
copyfile / = exec fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

5. Invoke CPYSVSES EXEC:

```
cpysvses exec lvlses ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. CPYSVSES EXEC invokes LVLSES, which invokes FILELIST to display each entry for each service EXEC listed in CPYSVSES. Process each FILELIST screen as follows:

   a. For each of the files on the FILELIST screen, determine which is the latest version. **Do not depend on the date and time stamps;** examine the actual files. Each file ends with an update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn*, MOD*nnnnn*, XED*nnnnn*, and PRF*nnnnn* are exact images of the corresponding EXEC, MODULE, XEDIT, and PROFILE executables.

   b. Copy the latest files to the task disk (5E5), using the OLDDATE REPLACE options. Assign the copy a filetype of EXEC, MODULE, XEDIT, or PROFILE:

```
copyfile / = ft fm (olddate replace ■
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

   c. Press **PF3** to exit this FILELIST screen and go to the next screen.

   d. Continue until there are no more FILELIST screens.

7. Look for the latest version of the product parameter file:

```
filelist 56643089 * * ■
```

Examine each version of the product parameter file. A sample product parameter file is shown in "A Sample Product Parameter File" on page 384. The product parameter file has an update history at the top. The latest level is the one with the most updates. Use the update history, **not the date and time stamp**, to determine which is the most recent version.

8. Copy the latest level of the product parameter file to the task disk (5E5):

```
copyfile / = $ppf fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy.

9. Now that the service tools are all on the service tools build disk, you do not need to issue SETUP again. Instead, simply access 5E5 as B. You may wish to add this access to MAINT's PROFILE EXEC.

```
access 5E5 b ■
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase any duplicate files from MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

filelist * * b ■	This command lists all the files on your B-disk.
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.
erase ■	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the B-disk (5E5) or any other disk on which it may appear.

Step 9. Set the System Default National Language

In this step, you will set the system default national language.

This step takes approximately 20 minutes.

The system default national language is mixed-case American English. This means that most system messages and HELP files will appear in mixed case (capital and small letters). Some will appear in uppercase (capital letters only).

Decide whether you want to keep the system default as mixed-case American English or change it to uppercase American English (in which case all system messages and HELP files will be in uppercase). Follow the appropriate instructions below.

If you want to install a system default national language other than American English, you will be able to do so after you finish the installation process. See Chapter 5, "Installing a New System National Language" on page 311 for instructions.

If you want to install an alternate system national language (including whichever version of American English is not your default), see *VM/XA SP Planning and Administration.*

Mixed-Case American English

If you want to keep the system default national language as mixed-case American English, you should convert the installation messages to mixed case. (If you do not do so, all the installation messages will continue to appear in uppercase, but most other system messages will appear in mixed case.)

1. Save the uppercase installation message file:

```
rename $msg4i$ exec a = excuceng = ■
READY; T=n.nn/n.nn hh:mm:ss
```

2. Rename the mixed-case installation message file:

```
rename $msg4i$ excameng a = exec = ■
READY; T=n.nn/n.nn hh:mm:ss
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSAMENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $MSG4I$ EXEC, the service will be in uppercase. You do not have to convert it to mixed case.

Uppercase American English

If you want to change the system default to uppercase American English, you must:

1. Edit the product parameter file and change all :NLS. tags from AMENG to UCENG:

```
xedit 56643089 $ppf ■
ch/:NLS.      AMENG/:NLS.      UCENG/* * ■
file ■
```

There are six blanks after :NLS. in the XEDIT change command.

2. Save DMSNGP TEXT, then rename the DMSNGP TXTUCENG file DMSNGP TEXT. This gives you a TEXT file assembled from a DMSNGP profile with LANGID = UCENG and HELP = 19C.

   Note: You will be able to make other changes in the DMSNGP profile in Step 13, after you install Assembler H. You will have to change the LANGID= and HELP= parameters in Step 13. The DMSNGP TXTUCENG file is provided here only to allow you to specify uppercase American English as your system default national language.

```
access 395 d ■
DMSACC724I 395 REPLACES D(395)
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp text d = txtameng = ■
READY; T=n.nn/n.nn hh:mm:ss
rename dmsngp txtuceng d = text = ■
READY; T=n.nn/n.nn hh:mm:ss
```

3. Convert $VMFMSG$ EXEC (the service message file) to uppercase:

```
xedit $vmfmsg$ exec ■
uppercas * ■
file ■
```

Notes:

1. You may see the following message when you IPL CMS:

```
DMSINS283E The NLSUCENG saved segment could not be found;
            return code 44 from SEGMENT
```

   This is an informational message. You can ignore it.

2. If you receive service for $VMFMSG$ EXEC, the service will be in mixed case. You will have to convert it to uppercase. If you receive service for the DMSNGP profile or the product parameter file, you will have to update them again.

Step 10. Build CMS

In this step, you will build CMS.

This step takes approximately 5 minutes.

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. IPL 190 and issue ITASK BUILD CMS to invoke the CMS BUILD process:

```
ipl 190 clear ■
VM/XA STARTER SYSTEM
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
itask build cms ■
nnnnnnnn files changed
⋮
DMSACC724I 191 replaces A(191)
DMSACC724I 5E5 replaces B(5E5)
```

ITASK invokes the SETUP EXEC to create the correct CMS access order.

```
DMSWSU1900W The existing 56643089 $SETUP A1
             file has been refreshed.  You
             might want to check your access
             order when done.
```

You will see this message only if you have a copy of 56643089 $SETUP on your A-disk. If you do see it, you will get return code 4.

```
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
PUN FILE fileno TO XAMAINT
  COPY 001 NOHOLD
```

A temporary load list is created and the system loader is invoked.

```
DMSWSU1906E The access of 19E failed with a
             return code of 28.  Processing
             stops for product 56643089.
             The original access order cannot
             be restored.
```

You may receive this message. It is not a problem. The access failed because 19E is empty. You will reaccess 19E when you re-IPL 190. If you see this message, you will get return code 4.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in XAMAINT's virtual reader:

```
query rdr xamaint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 8 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * hold ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The HOLD option causes the spool file to remain in your reader after it has been received. To remove this file from your reader, issue **purge rdr** *fileno*. To change the status of the reader file, issue **change rdr** *fileno* **hold**. *fileno* is the file number of the reader file.

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
HCPLDR8013I Possible overlay:  .SLC 000000
DMSINS283E The NLSlangid saved segment could not be found;
            return code 44 from SEGMENT
```

If you see either of these messages, you can ignore them. They are informational messages.

```
DMSINQ609R Nucleus (CYL or BLK) address =
cyl ■
```

Enter the correct starting cylinder address for the DASD type of your 190 minidisk:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

```
DMSINQ612R Enter version identification:
■
```

Press **ENTER** for the default version identification, VM/XA CMS 5.6 *mm/dd/yy hh:mm*. You will be able to modify the default in Step 13.

```
DMSINQ612R Enter installation heading:
■
```

Press **ENTER** for the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM. You will be able to modify the default in Step 13.

```
DMSINS327I The installation DCSS could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

Informational message: The optional installation segment (CMSINST) is not loaded and saved until "Step 31. Create the CMSINST and HELP Saved Segments" on page 301 is completed.

9. Log off to bring the VM/XA CMS nucleus on line.
10. Log on as XAMAINT:

```
logon xamaint ■
ENTER PASSWORD
password ■
```

Use the password defined in your **VM/SP** or **VM/SP HPO** directory.

```
DMKLNM108E XAMAINT 124 not linked; volid XAP001 not mounted
DMKLNM108E XAMAINT 125 not linked; volid XASERV not mounted

DMKLNM108E XAMAINT 126 not linked; volid XAP002 not mounted
```

You will see this message only if you are using 3350 DASD.

```
DASD 390 linked R/O, R/W by MAINT, R/O by OPERATOR
LOGON AT hh:mm:ss EDT day mm/dd/yy
FILES: nnn RDR, nnn PRT, nnn PUN

DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
```

Step 11. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to XAMAINT's virtual printer. Ready the printer to send the CMS load map as a printer file:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt ■
```

2. Identify the printer file and transfer it to your reader:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
XAMAINT  fileno A PRT 00006961 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Identify the reader file:

```
query rdr * all ■
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,100 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

4. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft d ■
File fn ft D received from XAMAINT at HPO 5.6
  sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

5. To print a copy of the CMS load map, issue:

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Examine the load map for errors:

```
xedit fn ft ■
:
qquit ■
```

7. Issue:

```
set ecmode on ■
CP ENTERED; DISABLED WAIT PSW '00020000 00000000'
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
READY; T=n.nn/n.nn hh:mm:ss
```

8. Continue with the next step.

Step 12. Install Assembler H Version 2 Program Product

Note: IBM packages Assembler H Version 2 as a separate program product. This program product has its own installation procedure.

In this step, you will:

- Consult *Assembler H Version 2 Program Product: Installation*, SC26-4030
- Install Assembler H Version 2 Program Product, VM feature
- Install service for Assembler H Version 2.

The time to perform this step may vary, but a good estimate is 20 minutes.

1. Before you install Assembler H, consult your IBM Support Center for the latest information on APARs.
2. Read the *Installation* manual for Assembler H Version 2 Program Product.
3. Refer to "Installing Program Products" the *VM/XA SP2.1 Program Directory* for more information on Assembler H.
4. Follow the directions in the *Installation* manual to install Assembler H Version 2. IBM advises you to install Assembler H Version 2 on the 19E minidisk.

   Note: Use the OBJECT option to assemble Assembler H CSECTS.
5. During the installation of Assembler H Version 2, answer YES to the question "BUILD AN AUXILIARY DIRECTORY?" and respond Y to the question asking what mode letter you want to use to access the disk containing the H-Assembler modules.
6. Erase IEV61 TEXT. If you do not erase this file, you will not be able to apply service to Assembler H.
7. Install the latest service level of Assembler H.
8. After you have finished installing service for Assembler H, IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Check MAINT 191 (your A-disk) for files with the same filename as files on the task disk (5E5). Erase any duplicate files from MAINT 191 (your A-disk) so that you will not invoke any back-level service EXECs.

`filelist * * fm ■`	This command lists all the files on the task disk (5E5).
PF9	Pressing **PF9** on the FILELIST screen lists all the files with the same filename as the one where the cursor is positioned, with any filetype and on all disks. See if there is one with the same filename on the A-disk.

`erase ■`	Enter this command on the line listing the file you want to erase. Be sure to erase the file only on the A-disk (191), not on the task disk (5E5).

10. If any unneeded service memos exist on the MAINT 191 disk, erase them to make room for further processing.

Step 13. Tailor the DMSNGP Profile

In this step, you will use the VM System Product Editor (XEDIT) to tailor the CMS nucleus generation profile (DMSNGP).

This step, excluding time for printing system definition files, takes approximately 15 minutes.

1. Before proceeding with this step, read the discussion of the DMSNGP profile in *VM/XA SP Planning and Administration*, under the discussion of the DEFNUC macro. DMSNGP ASSEMBLE is a profile that contains CMS configuration defaults and responses to system prompts; these will become part of the CMS nucleus.

2. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Access the CMS LOCAL1 minidisk, where DMSNGP resides, as D:

```
access 395 d ■
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

In this example the DMSNGP file resides on the 395 disk.

4. Save a copy of DMSNGP ASSEMBLE, so that if you make a mistake in changing it, you can go back to the original file:

```
copyfile dmsngp assemble d = oldassem = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. XEDIT the CMS nucleus generation profile:

```
xedit dmsngp assemble ■
```

Within the file, you can use the normal XEDIT cursor-control keys, scroll keys, insert key, and delete key.

```
NGP      TITLE 'DMSNGP     (CMS)      VM/XA SYSTEM PRODUCT 5664-308'
         EJECT
DMSNGP   CSECT
         DEFNUC  SYSDISK=190,         * S disk address                *
                 YDISK=19E,           * Y-disk address                *
                 HELP=19D,            * Help disk address             *
                 LANGID=AMENG,        * Default is American English   *
                 DBCS=NO,             * Default is not a DBCS lang    *
                 LANGLEV=S,           * DCSS ID for multiple DCSS     *
                 SAVESYS=NO,          * Using CMS in DCSS YES OR NO   *
                 SYSNAME=CMS,         * Name of above DCSS to save    *
                 USEINST=YES,         * Using EXEC/XEDIT in DCSS      *
                 INSTSEG=CMSINST,     * Name of above DCSS to save    *
                 REWRITE=YES,         * Write nucleus yes or no       *
                 IPLADDR=190,         * Address of where to write     *
                 CYLADDR=?,           * CYL/BLK OF WHERE TO WRITE     *
                 IPLCYL0=YES,         * Write IPL text on cyl 0       *
                 VERSION=?,           * VM/XA CMS 5.6 MM/DD/YY HH MM SS*
                 INSTID='VM/XA CONVERSATIONAL MONITOR SYSTEM'
         END
```

6. Look at the DEFNUC defaults, and make any necessary changes. Refer to *VM/XA SP Planning and Administration* for information on the DEFNUC macro.

   Warning: Do not delete the comma at the end of each line of the DEFNUC macro but the last line. If you do, all the lines after the first line without a comma are ignored.

   Warning: On each line, make sure that you leave at least one blank between the comma and the comment.

   - At a minimum, you should change **CYLADDR=?**, to indicate the starting cylinder on the MAINT 190 minidisk at which to start writing the CMS nucleus. The appropriate starting cylinder depends on the device type of the DASD where the MAINT 190 minidisk is defined:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

     If you do not replace the question mark for the CYLADDR keyword, you receive the following prompt when you generate the CMS nucleus:

     ```
     DMSINI609R Nucleus (CYL or BLK) address =
     ```

     You must respond with the appropriate cylinder address.

   - If you specify **YES** after the SAVESYS option, you must create a skeleton named saved segment (NSS) before building the CMS nucleus. This segment is saved when you build CMS and IPL your reader. (You will get message HCPNSS440I.)

     A skeleton segment for the CMS nucleus is created by the SAMPNSS EXEC. You can also create your own skeleton segment with the DEFSYS command. For information on the DEFSYS command, see *VM/XA SP CP Command Reference* and *VM/XA SP Virtual Machine Operation.*

- Make sure the LANGID and HELP parameters are right for your system default national language:

Language	LANGID	HELP
Mixed-case American English	AMENG	19D
Uppercase American English	UCENG	19C

- Make any other changes you wish. For example, you may want to change the version identifier or the installation header. You must put **single quotes** around your version identifier and installation header. The character string between the single quotes can be any combination of alphanumeric characters.

```
file = = fm ■
```

Make your changes and file the update file on the CMS LOCAL1 (395) disk.

7. Assemble the new DMSNGP file:

```
itask assemble dmsngp ■
⋮
DMSUPD181E No update files were found
DMSWHM1907I Assembling DMSNGP
```

The minidisk access order is reestablished.

Note: If an error occurs during the assembly, examine the flagged statements and correct the statements in error. Then continue from the beginning of this step.

```
DMSWHM1909I DMSNGP TEXT A created
```

The DMSNGP file has been assembled and placed on the MAINT 191 minidisk.

8. Copy the text deck from MAINT 191 to MAINT 395, then erase it from MAINT 191:

```
copy dmsngp text a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase dmsngp text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the filemode of the CMS LOCAL1 minidisk.

If you did not replace the question mark for the CYLADDR keyword, when you generate the CMS nucleus, you will receive the following message:

Message	Appropriate Response
DMSINI609R Nucleus (CYL or BLK) address =	*nnnnn* *nnnnn* is the location on the 190 disk where the new CMS nucleus is written. Refer to the table on page 245 to determine the correct response for your DASD type.

Step 14. ITASK EXEC Rebuilds the CMS Nucleus

Note: Do this step only if you changed the DMSNGP file in Step 13.

In this step, the ITASK EXEC:

- Creates a CMS nucleus
- Places the CMS nucleus on the CMS residence disk.

This step takes approximately 5 minutes.

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Issue ITASK BUILD CMS to create the CMS nucleus. By issuing the ITASK EXEC, you will incorporate updated CMS text files into a new CMS nucleus.

```
itask build cms ■
nnnnnnnn files changed
⋮
```

ITASK invokes the SETUP EXEC to create the correct CMS access order.

```
DMSWSU1900W The existing 56643089 $SETUP A1
             file has been refreshed.  You
             might want to check your access
             order when done.
```

You will see this message and return code 4 only if you did not erase 56643089 $SETUP in Step 10.

```
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

PUN FILE fileno TO XAMAINT
  COPY 001 NOHOLD

Ready; T=n.nn/n.nn hh:mm:ss
```

A temporary load list is created, and the system loader is invoked.

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in XAMAINT's virtual reader:

```
query rdr xamaint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
XAMAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file you will IPL in substep 8 below. It has approximately 28,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * hold ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The HOLD option causes the spool file to remain in your reader after it has been received. To remove this file from your reader, issue **purge rdr** *fileno*. To change the status of the reader file, issue **change rdr** *fileno* **hold**. *fileno* is the file number of the reader file.

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
HCPLDR8015I POSSIBLE OVERLAY:  .SLC 000000
```

You may receive this message. It can be ignored.

CMS Nucleus Generation Prompts and Responses

Note: Each of the following prompts appears **only** if the corresponding statement in DMSNGP is missing or empty (and the DEFNUC macro contains no default value), or if the DMSNGP statement contains a question mark (?):

```
DMSINQ606R System disk address =
■
```

190 is the default.

```
DMSINQ615R Y-disk address =
■
```

19E is the default.

```
DMSINQ640R HELP disk address =
■
```

19D is the default.

DMSINQ764R Language id =

■ or *langid* ■

This response identifies the *langid* of your system national language. The default *langid* is AMENG (mixed-case American English).

DMSINQ293R Is this a DBCS language? Enter 1 (YES) or 0 (NO).

0 ■

The default is 0 (NO); mixed-case American English is not a DBCS (Double-Byte Character Set) language.

DMSINQ295R Language level id =

■

The system national language does not use a level ID.

DMSINQ296R Should the installation segment be used? Enter 1 (YES) or 0 (NO).

The installation segment is an optional shared saved segment, into which you can place frequently used EXECs and System Product Editor (XEDIT) macros. You install the segment **after** you install your base system, but you must indicate now whether or not you are going to use it.

■ (or 1 ■) or 0 ■

The default is 1 (YES). Enter **0** if you do not want to use the segment.

DMSINQ310R Installation segment name =

This prompt appears only if you accepted the default (or entered **1**) at the previous prompt.

■ or *segname* ■

Enter a 1- to 8-alphanumeric-character name for the installation segment, or press **ENTER** to accept the default name, CMSINST.

DMSINI729R Do you want to save the system? Enter 1 (YES) or 0 (NO).

0 ■

Answer **0** (NO). You will save CMS in Step 29.

The default is 1 (YES). **Warning:** The default response to this prompt does not agree with the sample DMSNGP file, where SAVESYS=NO.

DMSINI730R Saved system name =

You will not see this prompt if you answered **0** to the preceding prompt.

■ or *sysname* ■

The default system names are CMS and CMSXA.

DMSINI607R Rewrite the nucleus? Enter 1 (YES) or 0 (NO).

1 ■

Enter **1** to write the CMS nucleus on the disk that you specify in your response to the next prompt.

DMSINI608R IPL device address =

■

The default is the address of the system disk (190).

DMSINI609R Nucleus (CYL or BLK) address =

nnn ■

nnn is the location on the 190 system disk where the new CMS nucleus is written. Enter the correct cylinder/block address for your DASD type:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

DMSINI610R Also IPL CYL/BLK 0? Enter 1 (YES) or 0 (NO).

1 ■

Enter **1** to write the initial IPL text on cylinder/block 0 of the disk where the CMS nucleus is written.

The initial IPL text is a bootstrap program that reads the CMS nucleus from the cylinder/block where the nucleus is written (as defined in your response to prompt DMSINI609R). The initial IPL text is always written on the same cylinder/block as the nucleus. If the initial IPL text is not also written on cylinder/block 0, you must specify the cylinder/block address of the nucleus when you issue IPL commands for this system. For more information, refer to the description of the IPL command in *VM/XA SP CP Command Reference*.

DMSINI611R Enter version identification:

The version identification is displayed each time that you IPL the CMS system you are now generating.

■ or *version* ■

You can enter up to 32 descriptive characters to identify this version and level of CMS, or you can press **ENTER** to accept the default version identifier, VM/XA CMS 5.6 *mm/dd/yy hh:mm*.

DMSINQ612R Enter installation heading:

The installation heading appears at the beginning of each output file created using this CMS nucleus.

■ or *heading* ■

You can enter up to 64 descriptive characters to serve as an installation heading, or you can press **ENTER** to accept the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM.

End of CMS Nucleus Generation Prompts and Responses

```
DMSINS327I The installation saved segment could not be loaded
```

Informational message: The CMSINST installation segment is not loaded and saved until you complete Step 29 on page 289.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■

⋮
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

This is the default version identification. If you defined your own version identification, it appears here and each time that you IPL 190 or IPL CMS.

Step 15. Save and Print the CMS Load Map

In this step, you will:

- Save a load map of CMS on the CMS LOCAL1 disk
- Print a copy of the load map.

This step takes approximately 5 minutes.

1. The CMS load map has been spooled to XAMAINT's virtual printer. Ready the printer to send the CMS load map as a printer file:

```
spool prt * nohold ■
Ready; T=n.nn/n.nn hh:mm:ss
close prt ■
```

2. Identify the printer file and transfer it to your reader:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
XAMAINT  fileno A PRT 00006961 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Identify the reader file:

```
query rdr * all ■
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,300 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

4. Bring in the file from the virtual reader to the CMS LOCAL1 disk:

```
access 395 d ■
Ready; T=n.nn/n.nn hh:mm:ss
receive fileno fn ft d ■
File fn ft D received from XAMAINT at HPO 5.6
  sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS map is loaded on the CMS LOCAL1 disk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1−9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

5. To print a copy of the CMS load map, issue:

```
spool 00e to system class a ■
Ready; T=n.nn/n.nn hh:mm:ss
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Examine the load map for errors:

```
xedit fn ft ■
locate /TXT ■
:
qquit ■
```

7. Continue with the next step.

Step 16. Convert the DMKRIO and DMKSYS ASSEMBLE Files and Tailor HCPBOX ASSEMBLE and the Product Parameter File

In this step, you will:

- Convert your current system's real I/O configuration and system definition sample files to files for your new system
- Tailor the HCPBOX ASSEMBLE file if you wish to use your own logo instead of the default logo that appears on your screen when you log onto the system or the default logo that appears on separator pages
- Assemble the HCPRIO, HCPSYS, and HCPBOX files
- Tailor the product parameter file only if you have a reason for changing the supplied CP nucleus generation information.

The time to perform this step varies depending upon the time you spend editing files.

Convert the DMKRIO and DMKSYS ASSEMBLE Files

Note: If you are applying updates with AUX files, follow the service procedures in Part 2, "Servicing the System."

Your current system's DMKRIO ASSEMBLE and DMKSYS ASSEMBLE files define the unique configuration of your current system. You must convert them to files defining your new system:

- The real I/O configuration file (HCPRIO ASSEMBLE) defines the configuration of your system input/output devices.
- The CP system control file (HCPSYS ASSEMBLE) describes the system residence device (XASRES) and various system parameters.

For full instructions on converting these files, see the *VM/XA SP Conversion Notebook.*

Note: You may find it useful to print copies of the system files for future reference. These include HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and your user directory (default USER DIRECT).

The sample HCPRIO and HCPSYS were loaded to the XAMAINT 295 minidisk. Your new files should also reside there.

Note: When tailoring HCPRIO, minidisk caching is not allowed for shared DASD. To control sharing on a device level, use the RDEVICE SHARED=YES option in HCPRIO or use the SET SHARED ON FOR *rdev* command. These will effectively turn off minidisk caching on a device basis. We recommend that you set the RDEVICE macro in HCPRIO rather than using the SET SHARED ON FOR *rdev* command.

When you convert DMKSYS ASSEMBLE to HCPSYS ASSEMBLE, be sure to compare it with the sample HCPSYS ASSEMBLE file. Load the VM/XA System Product Release 2.1 nucleus to the XASRES volume. After you have installed and tested VM/XA System Product, you can move the nucleus to the VM/SP system's SYSRES volume; but you must reformat and reallocate the CP areas first. See the *VM/XA SP Conversion Notebook* for details.

Tailor HCPBOX ASSEMBLE

If you want to change the design or contents of either the logo that appears when you log on to the system, or the logo that appears on separator pages in printed output, tailor the sample HCPBOX ASSEMBLE file. **Do not edit HCPBOX ASSEMBLE.** IBM services this file and bases the service on the version supplied. Use the procedure for local modifications.

1. Establish the appropriate minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you receive a return code of 4 and message DMSWSU1900W. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
vmfsetup 56643089 cp (all ■
erase 56643089 $setup a ■
```

2. Verify that HCPXA CNTRL lists an auxiliary control file for local changes:

```
xedit hcpxa cntrl ■
TEXT MACS HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT  AUXPAT TX$  * LOCAL PATCHES
LCL  AUXLCL LCL  * LOCAL MODIFICATIONS
TEXT AUXXA       *
qquit ■
```

HCPXA CNTRL should have a line like this one. The second field is the filetype of the auxiliary control file that points to local modifications.

3. Edit the auxiliary control file:

```
xedit hcpbox auxlcl ■
```

```
input ■
```

You will add your local modifications at the top of the file.

```
R00001LC LCL LC00001  *UPDATE HCPBOX LOGO ■
```

In this example. **R00001LC** is the filetype of the update file, and **LC00001** is the local tracking number. The rest of the line, beginning with the asterisk (*), is a comment explaining the purpose of this modification. The text deck created when you assemble HCPBOX with your local modifications will be called HCPBOX LCL00001.

```
■
```

Press **ENTER** to return to the command line.

```
file = = fm■
```

fm is the CP LOCAL1 disk (295).

4. Create the update file and apply the updates:

```
xedit hcpbox assemble (ctl hcpxa ■
```

Instead of editing HCPBOX ASSEMBLE directly, you will apply changes to a copy of it.

```
DMSXUP178I Applying HCPBOX Annnnn HP
```

The updates supplied by IBM for HCPBOX ASSEMBLE are being applied.

```
DMSXUP180W Missing PTF file HCPBOX R00001LC F5
```

Your update file cannot be found because you have not created it yet.

```
 HCPBOX R00001LC A1
      |...+....1....+....2....+....3....+
===== * * * Top of File * * *
=====    .
=====    .
=====    .

====>
```

The system copies the source file and allows you to edit the copy.

```
set serial off ■
set trunc 72 ■
⋮
file = = fm ■
```

You want the source file's sequence numbers and continuation characters to remain unchanged.

Make your changes and file the update file on the CMS LOCAL1 (295) disk.

Assemble the HCPRIO, HCPSYS, and HCPBOX Assemble Files

1. To assemble the HCPRIO, HCPSYS, and HCPBOX ASSEMBLE files, issue the following command for each file:

```
itask assemble filename ■
⋮
DMSWHM1907I Assembling filename
```

If an error occurs while assembling any of the system definition files, see "Correcting Assembly Errors."

```
DMSUPD178I Updating HCPBOX ASSEMBLE B1
DMSUPD178I Applying HCPBOX AnnnnnHP F5
DMSUPD178I Applying HCPBOX ft B5
DMSWHM1909I filename {TEXT|ft} A created
PRT FILE fileno TO XAMAINT
  COPY 001 NOHOLD
Ready; T=n.nn/n.nn hh:mm:ss
```

You see these messages only when you assemble HCPBOX ASSEMBLE, and only if you updated it.

2. Copy these text files from your A-disk to the CP LOCAL1 minidisk:

   HCPSYS *ft*
   HCPRIO *ft*
   HCPBOX *ft*

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
copy filename {text|ft} a = = fm (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase filename {text|ft} a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the CP LOCAL1 minidisk (295).

Correcting Assembly Errors

To correct assembly errors in system definition ASSEMBLE files, do the following:

1. Examine the flagged statements. See *VM/XA SP Planning and Administration* for directions on coding.
2. Correct the statements causing an error by editing the system definition file with the System Product Editor (XEDIT).
3. Reassemble the system definition file in which the change was made using the procedure in "Assemble the HCPRIO, HCPSYS, and HCPBOX Assemble Files."
4. Repeat the sequence until all files assemble correctly.

Tailor the Product Parameter File

The default 56643089 $PPF should be appropriate for most installations, but if your experience suggests changes, you can change 56643089 $PPF. (See "A Sample Product Parameter File" on page 384.)

DO NOT alter this file unless you have a special need.

The best way to change the product parameter file is to create an override file. To do this, see "The Product Parameter Override File" on page 402.

| Use the XEDIT command to create an override file on the TASK (5E5) minidisk. See the *VM/XA SP System Product Editor User's Guide* for information about the System Product Editor.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

| File the override file on the TASK (5E5) minidisk. Use the XEDIT subcommand FILE.

Step 17. Generate the New CP Nucleus

In this step, you will use the ITASK EXEC to build your new CP nucleus.

This step takes approximately 20 minutes.

The dialogs shown here are for sample use only. If you find a discrepancy between these dialogs and those you have on your terminal, please refer to the Program Directory that you received with the product tape for the most current information.

1. Purge all unnecessary files from your reader to increase available spool space.
2. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the ITASK EXEC to build the CP nucleus:

```
itask build cp noassem ■
```

Use the **noassem** only if you have already assembled HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and HCPBOX ASSEMBLE. If you have not assembled these files, you **must** omit the **noassem** operand, so that ITASK will assemble them now.

If you assemble the files here, copy the text decks from XAMAINT 191 to the CP LOCAL1 minidisk (295), then erase them from XAMAINT 191.

```
nnnnnnnn FILES CHANGED
DMSACC724I 191 replaces A(191)
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

PUN FILE fileno TO XAMAINT COPY 001 NOHOLD
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The CP nucleus is being created. It will be sent to your reader.

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CP, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

4. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

5. Ensure that the proper files are in XAMAINT's virtual reader:

```
query rdr xamaint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
XAMAINT  nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 10 below. It has approximately 75,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

6. Order your reader so that the CP nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CP nucleus.

7. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If the virtual reader is not class *, issue:

```
spool rdr class * hold ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The HOLD option causes the spool file to remain in your reader after it has been received. To remove this file from your reader, issue **purge rdr** *fileno*. To change the status of the reader file, issue **change rdr** *fileno* **hold**. *fileno* is the file number of the reader file.

9. Define your virtual storage as 16 megabytes. This is the maximum available on your existing VM/SP or VM/SP HPO system and the minimum needed to IPL your virtual reader in substep 10.

   Note: If you are generating a CP nucleus with a preferred virtual machine, XAMAINT's virtual storage must be at least 4 megabytes greater than the sum of the VRSIZE, VRFREE, and RIO370 operands in the SYSSTORE macro instruction in HCPSYS ASSEMBLE. Since you cannot have more than 16 megabytes of virtual storage in System/370 mode, the sum of the VRSIZE, VRFREE, and RIO370 operands must not be greater than 12 megabytes. You can make the sum larger after you initialize the system. For more information on the SYSSTORE macro instruction, refer to *VM/XA SP Planning and Administration.*

```
define storage 16m ■
STORAGE = 16384K
DMKDSP450W CP entered; disabled wait
           PSW 00020000 00000000
```

10. Load (IPL) XAMAINT's virtual reader:

```
ipl 00c clear ■
```

```
hh:mm:ss * MSG FROM XAMAINT :
HCPGEN9010W NUCLEUS LOADED ON XASRES
DMKDSP450W  CP entered; disabled wait
            PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus was successfully loaded on the system residence device.

If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

Step 18. Save and Print the CP Load Map

In this step, you will:

- IPL CMS
- Save the load map of the new CP system
- Print a copy of the new CP load map.

This step takes approximately 10 minutes.

1. When you built CP, the CP load map was spooled to MAINT's virtual printer. The load map must be transferred to MAINT's virtual reader. If you have IPLed 190 since building CP, the load map may have already been spooled from the virtual printer to the virtual reader.

   Query the printer to see if the load map is still there. If it is, note the file number. If it is not, skip substep 3 and go to substep 2.

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
MAINT  fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

   The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

2. IPL CMS:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

CP Load Map in Virtual Printer Only

3. Transfer the printer file to your reader:

```
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

End of CP Load Map in Virtual Printer Only

4. Access the CP LOCAL1 disk:

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the disk on which you will receive the load map.

5. Bring in the file from the virtual reader to the CP LOCAL1 disk:

```
receive fileno fn ft d (replace ■
fn ft D1 created
DMSRDC738I Record length is 132 bytes
File fn ft D received from XAMAINT at HPO 5.6
  sent as (none) (none) D
Ready; T=n.nn/n.nn hh:mm:ss
```

This command saves the load map on the CP LOCAL1 (295) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

6. To print a copy of the load map for the CP nucleus, issue:

```
spool 00c class a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

```
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The load map is printed on your system printer.

7. Examine the load map for errors:

```
xedit fn ft ■
⋮
```

8. Examine the load map for unresolved symbols. There should be no unresolved symbols.

```
locate /unresol ■
```

You are still in XEDIT mode.

```
DMSXDC546E Target not found
qquit ■
```

Step 19. Create the User Directory

In this step, you will:

- Create the user directory
- Use the DIRECTXA command to update the user directory.

The time to perform this step varies depending on the time you take to edit files.

1. Consult the *VM/XA SP Conversion Notebook* for instructions on converting the directory you used with your VM/SP or VM/SP HPO system.

   If you prefer to edit the sample directory supplied with your new system, a copy is printed in Appendix C, "VM/XA System Product Starter System Information" on page 725.

   Note: All logon passwords are NOLOG and there are no MDISK passwords, except for some of MAINT's minidisks. You may want to change these.

2. Edit your directory. (If you are using the sample directory instead of your old directory as a base, you must access 295 as B first. The sample directory, called USER DIRECT, is on MAINT's 295 minidisk.) Use the XEDIT command to edit the file.

   Make sure that none of the passwords in your directory are restricted. (For a list of restricted passwords, see Appendix F, "Restricted Logon Passwords" on page 859.)

   For more information about the user directory, see *VM/XA SP Planning and Administration.*

3. When your changes are complete, enter **file** on the command line.

4. Use the DIRECTXA command to update and place the user directory online:

```
directxa user direct ■
```

If your directory is not called USER DIRECT, enter your own directory name instead.

```
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED
Ready(0006); T=n.nn/n.nn hh:mm:ss
```

Since you are running on a VM/SP or VM/SP HPO system, the directory is not brought online.

Step 20. Put the VM/XA System Product Stand-Alone Dump Utility on Tape or DASD

In this step, you will put the VM/XA System Product stand-alone dump utility on tape or DASD.

Because VM/XA System Product stores dumps in system data files, you cannot transport them from your VM/XA System Product system to your VM/SP or VM/SP HPO system. If you have a problem and need to back off to VM/SP or VM/SP HPO, you will need the stand-alone dump facility to examine a dump of the problem.

Refer to *VM/XA SP Planning and Administration* for instructions on putting the stand-alone dump facility on tape or DASD.

Step 21. Prepare the CP-Owned DASD

In this step, you will:

- Reformat cylinder 0 of your CP-owned paging and spooling packs
- Reallocate the packs.

1. Shut down the system.
2. Re-IPL the system with the CP-owned paging and spooling packs offline.
3. Attach the CP-owned DASD to a virtual machine.
4. Use the CPFMTXA module to reformat only cylinder 0 of each pack.
5. Reallocate each pack.

For more information, see the *VM/XA SP Conversion Notebook.*

Step 22. Migrate Spool Files using the SPTAPE Command

In this step, you will use the SPTAPE command to migrate spool files from your current system to your new system.

You can migrate **spool files**, but not **system files**, from VM/SP or VM/SP HPO to VM/XA System Product. Use the CP SPTAPE command to do this.

Note: When you migrate a spool file from VM/SP to VM/XA System Product, it loses those attributes that have no VM/XA System Product equivalents. Therefore, if you migrate it back, it will no longer be identical to the original file.

See *VM/XA SP Conversion Notebook* for more information.

Step 23. Load the New CP Nucleus

In this step, you will:

- Shut down your VM/SP or VM/SP HPO system
- Re-IML your processor to 370-XA mode.
- Perform a hardware initial program load of the new CP nucleus and IPL VM/XA System Product with a cold start.

This step takes approximately 10 minutes.

1. Shut down the system you are now using. A class A user (usually the primary system operator) must issue the SHUTDOWN command.

```
shutdown ■
SYSTEM SHUTDOWN STARTED
Ready; T=n.nn/n.nn hh:mm:ss
SYSTEM WARM START DATA SAVED
SYSTEM SHUTDOWN COMPLETE
HCPGIR450W CP ENTERED; DISABLED WAIT
             PSW 000A0000 00000961
```

2. Re-IML the processor to 370-XA mode. Follow the hardware operations guide for your processor.
3. IPL the real address of your system residence device. Follow the hardware operations guide for your processor.

   If you are unable to IPL your system residence device, go back to "Step 2. Restore the VM/XA System Product Starter System to Disk" on page 217 and continue from there.

```
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;
SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss,
LOADED FROM XASRES
*****************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
*****************************************
⋮
HCPISU951I CP VOLID volid NOT MOUNTED
```

Note: A 9025 disabled wait state indicates that your system volume (XASRES) is not defined in the HCPRIO ASSEMBLE file. A 1010 disabled wait state indicates that your console is not defined in the HCPRIO ASSEMBLE file.

4. During the initialization phase, respond **cold drain** to the START message:

```
START ((COLD|WARM|FORCE) (DRAIN) (DISABLE) (NODIRECT))|(SHUTDOWN)
cold drain ■
```

```
NOW hh:mm:ss EDT day mm/dd/yy
CHANGE TOD CLOCK (YES|NO):
no ■
```

Note: If the clock is not set, set the TOD clock using standard operating procedures. Consult *VM/XA SP Real System Operation* for those procedures.

```
The directory on volume XASRES at address nnnn
   has been brought online.
There is no logmsg data
FILES: NO RDR, NO PRT, NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy
```

```
HCPCPJ951I CP volid volid not mounted
⋮
```

volid is a DASD volume listed on the SYSCPVOL macro instruction of HCPSYS. If the volume is not one that you are using, ignore the message.

You are now logged on to the OPERATOR user ID.

```
STORAGE = 0016M
FILES 0000001 RDR, 0000001 PRT, NO PUN
```

5. Define the storage that you want for the OPERATOR user ID:

```
define storage 16m ■
STORAGE = 16M
STORAGE CLEARED - SYSTEM RESET

XAUTOLOG AUTOLOG1
XAUTOLOG EREP
XAUTOLOG DISKACNT
```

This is an example. You may define more than 16M.

6. Enable the devices in your revised HCPRIO file:

```
enable all ■
hh:mm:ss Command complete
terminal mode vm ■
```

7. IPL 190:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

```
DMSACP1123E A(191) device error
Ready; T=n.nn/n.nn hh:mm:ss
```

If you see this error message, you have not formatted the operator's 191 minidisk.

8. Format the operator's 191 minidisk **only if you have not already done so:**

```
format 191 a ■
DMSFOR603R FORMAT will erase all files on disk A(191).
           Do you wish to continue?
           Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
opr191 ■
DMSFOR733I Formatting disk A
DMSFOR732I n cylinders formatted on A(191)
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Disconnect from OPERATOR and log on as MAINT.

   Note: If you do not have XAP001, you will receive this message:

   ```
   HCPLND108E MAINT 0124 NOT LINKED
   ```

   when you log on as MAINT. If you are using 3380 or 3390 DASD, you can ignore this message.

Correcting Load Errors

If the new CP nucleus fails to load, do the following:

1. IPL CMS:

```
ipl 190 clear ■
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If you have changed the version heading, your own heading will appear.

2. Access 295, then inspect the CP load map and virtual PSW. An error in loading may show up in either the load map or in the PSW.

   For a list of loader wait-state codes, refer to *VM/XA SP System Messages and Codes Reference.*

3. Correct the error.

4. Go back to the point where the error may have occurred. If you cannot determine where the error may have occurred, see "Step 12. Install Assembler H Version 2 Program Product" on page 242.

Step 24. ITASK Loads HELP Files from the Product Tape (Volume 2)

In this step, the ITASK EXEC loads HELP files from Volume 2 of the product tape.

The time this step takes varies from system to system, but a good estimate is 40 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 2 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD AMENGHLP, LOAD UCENGHLP, or LOAD HELP operands to load the HELP files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK command again.

```
itask load {amenghlp|ucenghlp|help} ■
```

The HELP files on the tape are in mixed-case American English:

- Use the **amenghlp** operand to load the HELP files to the 19D disk.
- Use the **ucenghlp** operand to load the HELP files directly to the 19C disk, then convert them to uppercase. Conversion takes about 20 minutes.
- Use the **help** operand to load the HELP files to the 19D disk, then copy them to the 19C disk and convert the copied files to uppercase.

```
DMSWSL409I Loading AMENGHLP FILES to the
           19D disk attached to MAINT
```

You will see this message if you used the **amenghlp** or **help** operand.

```
DMSWSL409I Loading UCENGHLP FILES to the
           19C disk attached to MAINT
```

You will see this message if you used the **ucenghlp** operand.

```
CONVERTING HELP FILES TO UPPERCASE
Ready; T=n.nn/n.nn hh:mm:ss
```

You will see this message if you used the **ucenghlp** or **help** operand.

Step 25. ITASK Loads Source Files from the Product Tape (Volume 3)

In this step, the ITASK EXEC loads CP, CMS, and dump viewing facility source files from Volume 3 of the product tape.

The time this step takes varies from system to system, but a good estimate is 30 minutes.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach your tape device as 181 and mount Volume 3 of the product tape on it.

   Follow the operation manual for the machine on which you mount the tape.

2. Enter the ITASK command with the LOAD ALL3 operands to load all the source files. Should a problem occur during this step, or for some reason you want to return to this step and load these tape files, you can invoke the load function by entering the ITASK LOAD ALL3 command again.

```
itask load all3 ■

DMSWSL409I Loading CP SOURCE to the
           394 disk attached to MAINT
DMSWSL409I Loading CMS SOURCE to the
           393 disk attached to MAINT
DMSWSL409I Loading DUMPVIEW SOURCE to the
           393 disk attached to MAINT

Ready; T=n.nn/n.nn hh:mm:ss
```

These are messages from ITASK as it loads files from the product tape.

ITASK has finished restoring source files from Volume 3 of the product tape.

Note: Source files are **not** unpacked during the load.

Step 26. Install Environmental Record Editing and Printing

In this step, you will:

- Read the documentation for the Environmental Record Editing and Printing (EREP) Program
- Follow the directions to install EREP.

The time to perform this step may vary, but a good estimate is 15 minutes.

Note: IBM packages the Environmental Record Editing and Printing (EREP) Program, VM Feature, 5654-260, as a separate program product. This program product has its own installation procedure.

1. Read the documentation for EREP:
   - The following memos in tape files 1 and 2 of the VMSUP EREP tape:
     - 5654260B MEMO
     - F5654260 MEMO2
     - I5654260 MEMO
     - 5654260B SERVICE
   - The Program Directory for EREP, which describes the installation procedure for EREP.

     Note: This document does not come with the VMSUP tape. You must order it separately.
   - The discussion of EREP in *VM/XA SP Planning and Administration.*
2. Follow the directions in the EREP *Program Directory* to install EREP from the VMSUP EREP tape.

   Note: You will receive messages HCPCRC8059I and HCPCRC8060I until EREP is installed.
3. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 27. Install the Printer Image Library

In this step, you will install the image library for your printer and save it on tape.

This step takes approximately 10 minutes.

Warning: Do not attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.

Note: VM/XA System Product handles the printer image library differently from VM/SP. For a discussion of the differences, see the *VM/XA SP Conversion Notebook*.

1. Load (IPL) 190. This is the CMS system disk.

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Establish the correct minidisk access order:

   Note: When you run VMFSETUP during the install procedures, it leaves the file 56643089 $SETUP on your A-disk. If VMFSETUP is run while this file exists on your A-disk, you receive a return code of 4 and message DMSWSU1900W. You may want to erase 56643089 $SETUP after running VMFSETUP.

```
access 5E5 b ■
vmfsetup 56643089 cp ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Invoke the IMAGELIB command. The command requires the filename of the image library (*fn* CNTRL). IBM supplies default image library control files called IMAG*xxxx* CNTRL, where *xxxx* is the printer number: 3800, 1403, 3203, 3211, 3262, 4245, or 4248.

   For more information about image libraries, see *VM/XA SP Planning and Administration*.

```
imagelib libname ■                         If you use the IBM-supplied image library,
HCPNMT247I NAMED IMAGE IMAGxxxx CREATED    libname is IMAGxxxx, where xxxx is the printer
Ready; T=n.nn/n.nn hh:mm:ss                number.
```

4. Check the status of the image library:

```
query img all ■
```

```
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*IMG     nnnn IMG  A  nnnn mm/dd hh:mm: fn       IMG      MAINT
```

Check the current state of the file (CL), the filename, and filetype.

5. Issue the appropriate LOADBUF command and start the printer. See *VM/XA SP CP Command Reference* for more information.
6. Mount a scratch tape with ring on an available 3420, 3422, or 3430 tape drive, or insert a cartridge into an available 3480 unit, with the thumbwheel turned until the white dot is not visible. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT. SPTAPE will attach it. If you have any other tape unit attached, detach it.
   - The tape is suitable for a 3420, 3422, 3430, or 3480 tape unit.

   Note: This procedure assumes the tape mode is 1600 bits per inch.
7. Issue the SPTAPE command to dump the image library to tape:

```
sptape dump vdevno img all run ■
```

Substitute the virtual device number of the tape drive for the value *vdevno*. The operand RUN specifies that SPTAPE rewinds and unloads the tape after the operation.

```
DUMPING vdevno IMG *IMG    nnnn A
⋮
SPTAPE DUMP FUNCTION ON
   DRIVE vdevno COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The message from SPTAPE tells you the file is being dumped to tape.

8. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE LOAD 181 IMG ALL RUN command to restore the image libraries. For more information about the SPTAPE command, see *VM/XA SP Real System Operation.*
9. If necessary, reattach your tape drive.

Step 28. Load, Build, and Save GCS

You must install the GCS (group control system) component if you plan to install SNA products or RSCS Version 2. **Before you begin, read "Planning for the Group Control System (GCS)" on page 9.**

If you do not want to install GCS, go to "Step 30. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments" on page 293.

To install GCS, you will do the following steps:

- Load GCS from Volume 2 of the product tape.
- Invoke the GROUP EXEC and complete a series of panels to create a GCS configuration file.

  Note: If you do not have a full-screen display device, you cannot use the GROUP EXEC because you cannot display the panels. In that case, you must build the configuration file manually, using the build macros described in the *VM/XA SP Group Control System Command and Macro Reference*.
- After you complete the panels, you invoke the ITASK EXEC with the BUILD GCS parameters.

  ITASK:
  - Modifies a copy of the GCS loadlist (GCSLOAD EXEC) and changes the entry that contains the filename of the GCS configuration file (the default is GCS) to match the filename of the configuration file that you just created (the default is also GCS).
  - Files the modified loadlist on the MAINT 295 minidisk. The modified loadlist is used during the generation of the GCS named saved system.
  - Renames the filetype of the GCS configuration file from GROUP to ASSEMBLE.
  - Invokes VMFHASM EXEC to assemble the configuration file.
  - Invokes VMFBLD EXEC with the BUILD GCS parameters to build and save the GCS nucleus.
- After you build and save the nucleus, you can save and print the GCS load map, which contains the storage addresses of the nucleus control sections and entry points.
- If you want to install more than one GCS nucleus, you must re-access the GCS minidisk structure and re-invoke the GROUP EXEC to create another GCS configuration file. Each configuration file must have a unique name and have an entry in the SAMPNSS EXEC.

This step takes approximately 30 minutes for each GCS system.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

1. Attach a tape drive to MAINT.
2. Mount Volume 2 of the product tape.

3. Issue ITASK LOAD GCS to load the GCS code:

```
itask load gcs ■
DMSWSL409I LOADING GCS INTERFACE TO
     THE 190 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS OBJECT TO
     THE 595 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS PUTDELTA TO
     THE 596 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORAPPLY TO
     THE 592 DISK ATTACHED TO MAINT
DMSWSL409I LOADING GCS CORDELTA TO
     THE 596 DISK ATTACHED TO MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

4. IPL 190:

```
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■                                          Press ENTER to complete CMS initialization.
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Establish the correct minidisk order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Set up the GROUP EXEC messages in storage:

   Notes:

   a. If you are REBUILDING a GCS nucleus and your current configuration file does not require changes, you can skip this part of the step and go directly to the nucleus build operation on page 284.

   b. If you do not have a full-screen display device, you **cannot** use the GROUP EXEC to build the configuration file. **Do not** set up the GCS messages in storage. Instead, refer to the *VM/XA SP Group Control System Command and Macro Reference* and use the build macros described there to build the configuration file manually. Then continue this step with the nucleus build operation on page 284.

```
copyfile csimes[y] [text|txtnnnnn] fm csiume[y]text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

y is the country code for your system default national language: blank for mixed-case American English and **b** for uppercase American English. You can use QUERY LANGLIST to determine your system default national language.

The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a field that reads UM*nnnnn*, where *nnnnn* is the PTF number.

Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*. You will have one with a filetype of TEXT.

This command creates a temporary GCS message file (containing the messages for the GROUP EXEC) that has the filename required by the SET LANGUAGE command. A permanent message file will be created when you generate the GCS nucleus. If there has been any service to the message file, the permanent file will reflect it.

```
set language langid (add csi user ■
Ready; T=n.nn/n.nn hh:mm:ss
```

langid identifies your system default national language. At this point, *langid* will be either **ameng** (mixed-case American English) or **uceng** (uppercase American English).

This command sets the language of the temporary GCS message file and places the file in user storage.

7. Invoke the GROUP EXEC to display the configuration panels.

   Use your installation reference worksheet to help you complete the panels. For more information, see "Planning for the Group Control System (GCS)" on page 9.

```
group systemname ■
```

This command assigns *systemname* as the filename of the GCS configuration file that you are creating and invokes the Primary Option Menu. The *systemname* parameter is optional at this time. If specified, it must match the system name in the DEFSYS entry for this GCS system in the SAMPNSS EXEC (the sample system name is GCS). If you specify a system name here, the Primary Option Menu appears with the system name filled in. If you do not specify a system name here, you must specify one on the Primary Option Menu.

Table 4. Function Keys Used with the GROUP EXEC Panels	
Key	**Function**
PF1 HELP	Shows formation about the panel you are looking at.
PF2 CLEAR	Clears the input areas where you enter information.
PF3 END	Leaves the present panel and returns you to a previous one. • If you press **PF3** on the Primary Option Menu, you return to CMS. • If you press **PF3** on any other screen, you return to the Primary Option Menu. • If you press **PF3** after you have entered information, but have not saved it, you receive the message: `577E File has been changed; type QQUIT to quit anyway`
PF5 REFRESH	Fills in the panel's input areas with the values you last saved there.
PF6 SAVE	Saves information you've entered on the panel (for the configuration file *systemname* GROUP).
PF7 PREVIOUS	Returns to the previous panel (if any).
PF8 NEXT	Moves ahead to the next panel (if any).
PF9 VERIFY	This PF key is not supported by VM/XA SP.
PF12 CURSOR	Moves the cursor to the panel's command line.
PF4, **PF10**, **PF11**	Not used.
ENTER	Processes any valid CP or CMS command typed on the command line. Two specific commands you can enter are: 1. QQUIT, entered from the Primary Option Menu, returns you to CMS. If entered on any other panel, it returns you to the Primary Option Menu. 2. CANCEL, entered on any panel, returns you to CMS.

```
 GRP1                  GCS GROUP - PRIMARY OPTION MENU                  Primary
-------------------------------------------------------------------------------


Fill in the blanks with the required information and then press the ENTER key.

       Type/change the name of the saved system that is being defined.

                       SYSTEM NAME : ________

          Type one number from the list below to display/update the:

                     1.  Authorized VM Userids.
                     2.  Saved System Information.
                     3.  Saved Segment Links.

             Type your choice here: _


-------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  ...      6  ...
  PF: 7  ...      8  ...      9  VERIFY   10  ...    11  ...     12  CURSOR

====>
```

8. To complete the Primary Option Menu:

   a. **Fill in or change the SYSTEM NAME.**

      If you invoked GROUP with the *systemname* parameter, this panel appears with the SYSTEM NAME already filled in.

   b. **Select the next panel (1, 2, or 3) and press ENTER.**

      The first time you leave the Primary Option Menu, select panel 1; the next time, panel 2; the third time, panel 3.

```
GRP11          AUTHORIZED VM USERIDS FOR <     >        PAGE 1 OF 1
------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the authorized VM userids.
       To CHANGE, type a new userid over the userid to be changed.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.


          ________   ________   ________   ________________
          ________   ________   ________   ________________
          ________   ________   ________   ________________
          ________   ________   ________   ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY  10  ...    11  ...      12  CURSOR

====>
```

9. To complete the Authorized Userids panel:

   a. **Type in the user IDs.**

      Press **ENTER** or **PF6** after typing the user IDs to save your information. Every time you press **ENTER** or **PF6**, the GROUP EXEC tells you how many user IDs it has processed. Use **PF7** and **PF8** if you have more than one screenful of user IDs.

   b. **Return to the Primary Option Menu.**

      Press **PF3**. If you forget to press **ENTER** or **PF6**, you will remain on this screen and see the message:

      ```
      577E File has been changed; type QQUIT to quit anyway
      ```

      To continue, simply press **PF6**. You will receive a message telling you how many authorized user IDs the EXEC has processed. Press **PF3** again to return to the Primary Option Menu.

```
GRP121            SAVED SYSTEM INFORMATION FOR  <     >       PAGE 1 OF 2
------------------------------------------------------------------------------


       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.

   SYSTEM DISK address (S-disk) . . . . .:        595
   SYSTEM DISK EXTENSION address (Y-disk):        59E

   USERID to RECEIVE STORAGE DUMPS. . . .:        ________
   RECOVERY MACHINE USERID (required) . .:        ________

   GCS TRACE TABLE SIZE (minimum 4K). . .:        ____16K


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5  REFRESH  6  SAVE
  PF: 7  ...      8  NEXT     9  VERIFY  10  ...    11  ...      12  CURSOR

====>
```

10. To complete the first Saved System Information panel:

    a. **Specify your disk addresses.**

       The default 595 and 59E virtual addresses are already saved and verified. If you must choose different addresses, type over the default values.

       Note: Make sure the new information is correct; the GROUP EXEC does not prevent you from saving invalid information.

    b. **Name an authorized dump receiver.**
       Name an authorized recovery machine.

       GCS requires you to enter a recovery machine user ID. The GROUP EXEC tells you whether that user ID is valid, but it does not prevent you from saving an invalid entry.

    c. **Specify a trace table size.**

       By default, the GROUP EXEC saves a value of 16K. If you decide to save a different amount, simply type the new value over the default value.

    d. **Save your input.**

       If you are satisfied with your choices, press **PF6** or **ENTER** to save the information.

       Warning: If you do not save the information before going on, the system will not accept it as input.

    e. **Go to the second panel.**

       Press **PF8** to continue on the second Saved System Information panel.

```
GRP122            SAVED SYSTEM INFORMATION FOR <      >        PAGE 2 OF 2
-------------------------------------------------------------------------------

       To ADD,    fill in the blanks with the information.
       To CHANGE, type the information over the displayed value.
       To DELETE, type blanks on the line.

          Pressing the ENTER key or PF6 key will save the update.

MAXIMUM NUMBER of VIRTUAL MACHINES (required). .:_____________________________

SYSTEM ID (maximum 130 characters) . . . . . .:_____________________________

_______________________________________________________________________________

_______________________________________________________________________________



-------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...      5  REFRESH  6  SAVE
  PF: 7  PREVIOUS 8  ...      9  ...     10  ...     11  ...      12  CURSOR

====>
```

11. To complete the second Saved System Information panel:

    a. **Specify a MAXIMUM NUMBER.**

       Simply type a value of 1 or more, and press **ENTER**. The GROUP EXEC responds with a message *only* if you enter an invalid value.

    b. **Type in your SYSTEM ID text.**

       This is optional. Move the cursor to the SYSTEM ID space (if it is not already there) and enter your text. The GROUP EXEC does not let you enter any more than 130 characters.

    c. **Save your input and return to the Primary Option Menu.**

       Press **ENTER** to save your information and then press **PF3**.

```
GRP13       AUTOMATIC SAVED SEGMENT LINKS FOR <      >    PAGE 1 OF 1
------------------------------------------------------------------------------


      To ADD,    fill in the blanks with the saved segment names
                 that will be linked automatically during
                 initialization of this virtual machine group.
      To CHANGE, type a new saved segment name over the saved
                 segment name to be changed.
      To DELETE, type blanks on the line.

         Pressing the ENTER key or PF6 key will save the update.

        ________    ________    ________    ________________
        ________    ________    ________    ________________
        ________    ________    ________    ________________
        ________    ________    ________    ________________


------------------------------------------------------------------------------
  PF: 1  HELP     2  CLEAR    3  END      4  ...     5   REFRESH 6  SAVE
  PF: 7  PREVIOUS 8  NEXT     9  VERIFY  10  ...    11   ...    12  CURSOR

====>
```

12. To complete the Automatic Saved Segment Links panel:

    Note: Do not include CMSVSAM and CMSBAM segments here.

    a. **Type in the segment names.**

       Press **ENTER** or **PF6** after typing each segment nameto save your information and advance to the next space at the same time. Every time you press **ENTER** or **PF6**, the EXEC tells you how many segment names it has processed. Use **PF7** and **PF8** if you have more than one screen of names.

    b. **Return to the Primary Option Menu.**

       If you have not saved your input yet, press **ENTER** or **PF6**. Press **PF3** to return to the Primary Option Menu.

13. Once you have provided the necessary input on all panels and returned to the Primary Option Menu for the last time, press **PF3** to exit from the GROUP EXEC.

14. Remove the temporary GCS message file from user storage and erase it on your A-disk:

```
set language langid (delete csi user ■
erase csiume[y] text a ■
```

langid is your system default national language. If you have not installed a different one, your default is **ameng**.

15. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

16. Invoke the ITASK EXEC with the BUILD GCS parameters:

```
itask build gcs systemname ■
```

systemname must match the name you specified when you issued the GROUP EXEC. If you do not specify a name, the default is GCS.

```
HCPNSD440I The Named Saved System (NSS) systemname
            was successfully defined
            in fileid fileno
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT

DMSUPD181E No update files were found
DMSWHM1907I Assembling systemname
DMSWHM1909I systemname TEXT A created
191 replaces A (191)
5E5 replaces B (5E5)
```

If You Receive an Assembly Error

If you receive an assembly error, you may have to reestablish the minidisk access order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■
```

Then go back and check your configuration file, which ITASK has renamed *systemname* ASSEMBLE. Refer to "Planning for the Group Control System (GCS)" on page 9 for information regarding required fields. Correct the file or go through the GROUP EXEC panels again to recreate the file. (If you recreate the configuration file, you must use the same *systemname*.) After you recreate the configuration file, rename or erase the old *systemname* ASSEMBLE file, then rename the new configuration file from GROUP to ASSEMBLE. Issue the ITASK BUILD GCS *systemname* command to assemble the configuration file and to build and save the nucleus.

End of If You Receive an Assembly Error

```
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
nnnnnnn FILES CHANGED
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

These are informational messages. You can ignore them.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The GCS nucleus has been sent to MAINT's virtual reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside GCS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

17. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

18. Copy the text deck, configuration file, ASSEMBLE file, and EXEC from MAINT 191 to MAINT 495:

```
access 495 d ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname text a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname text a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname group a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname group a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname assemble a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname assemble a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

19. Verify that the GCS nucleus is in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss fn  ft       SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 22 on page 286. It has approximately 8,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

The name of the IPL deck is the same as the last entry in the GCS load list. If you have difficulty identifying the IPL deck and need to know the reader file number, refer to the last entry in the GCS load list *systemname* EXEC. (The default name is GCSLOAD EXEC).

20. Order your reader so that the GCS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the GCS nucleus.

21. Query the reader. If it is not class *, spool it as class *:

```
query virtual c ■
RDR 000C CL cl NOCONT NOHOLD EOF READY
    000C nnnn                  CLOSED  NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
spool c class * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

22. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■

MSG FROM MAINT  :  CSIINI134I systemname has nnnnnn
   bytes of available common free storage
HCPNSS440I Named Saved System (NSS) systemname
            was successfully saved in fileid fileno

PRT FILE fileno SENT FROM MAINT PRT WAS fileno
  RECS nnnn CPY 001 NOHOLD NOKEEP
```

This message indicates that the GCS load map file has been sent to MAINT's virtual printer. If you plan to save or print the GCS load map, record *fileno*.

23. IPL 190:

```
ipl 190 ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: If necessary, purge your reader to free spool space.

Saving and Printing the GCS Load Map

24. The GCS load map has been sent to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME  NAME TYPE   DIST
MAINT    fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

The GCS load map is the file with a blank filename and filetype. It has approximately 2,200 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

```
#cp transfer prt fileno to * rdr ■
access 495 d ■
receive fileno fn ft d ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print fn ft d ■
```

End of Saving and Printing the GCS Load Map

Installing Multiple GCS Systems

25. You can install more than one GCS system, if you defined the additional systems in the system directory.

    To install each additional system:

    a. Reestablish the minidisk access order:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■
```

    b. Set up the GROUP EXEC messages in storage (full-screen display devices only):

```
copyfile csimes[y] [text|txtnnnnn] fm csiume[y]text a ■
```

    The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a field which reads UM*nnnnn*, where *nnnnn* is the PTF number.

    Note: If you have not received service on the CSIMES text file, you do not have a CSIMES file with a filetype of TXT*nnnnn*. You will have one with a filetype of TEXT.

```
copyfile csimes[y] text fm csiume[y] text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

```
set language langid (add csi user ■
```

    c. Create a configuration file for each nucleus.

    - If you are using a full-screen display device, issue:

```
group systemname ■
```

      systemname is a unique filename. Then complete the GCS panels to define the system parameters.

    - If you are **not** using a full-screen display device, refer to the *VM/XA SP Group Control System Command and Macro Reference*. Use the build macros to create the configuration file.

    d. Remove the temporary GCS message file from user storage and erase it from your A-disk (full-screen display devices only):

```
set language langid (delete csi user ■
erase csiume[y] text a ■
```

e. For each new GCS system, add a DEFSYS statement to the SAMPNSS EXEC. For information on the DEFSYS command, see the *VM/XA SP CP Command Reference*.

f. For each new GCS system, add an override section at the end of the product parameter file. This example shows an override section for a GCS system called **NEWGCS**:

```
:NEWGCS. GCS
:BLD. REPLACE
 NEWGCS VMFBDNUC BUILD1
:END.
```

You must also add the name of the new GCS system to the :OVERLST. tag at the top of the product parameter file, for example:

```
:OVERLST. CORCP CORCMS CORGCS NEWGCS
```

g. For each GCS system, assemble the configuration file, build and save the new GCS nucleus, and copy the text deck, configuration file, ASSEMBLE file, and EXEC to the 495 disk:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■


itask build gcs systemname ■
access 495 d ■
copy systemname text a = = d (replace ■
erase systemname text a ■
copy systemname group a = = d (replace ■
erase systemname group a ■
copy systemname assemble a = = d (replace ■
erase systemname assemble a ■
ipl 00c clear ■
ipl cms ■
```

Before you build another GCS nucleus, save or print the GCS load map, as indicated above. Remember to give each load map file a different name.

End of Installing Multiple GCS Systems

26. Verify that all your GCS segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss GCS         NSS      MAINT
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss systemname  NSS      MAINT
⋮
```

27. When you complete your GCS installation, go to "Step 29. Save CMS" on page 289.

Step 29. Save CMS

In this step, you will save CMS as a named saved system, in both 370 mode and 370-XA mode, at default addresses.

This step takes approximately 10 minutes.

Note: For general information about CMS as a named saved system, see *VM/XA SP Planning and Administration.*

1. Make sure:
   - You are logged on as MAINT.
   - You have finished loading all files to the 190 and 19E minidisks. After this step, any change to files on these disks requires resaving CMS.
   - You have 16 megabytes of virtual storage. To determine your storage size, issue QUERY VIRTUAL STORAGE. If you have less than 16 megabytes, issue DEFINE STORAGE 16M.
   - CMS is running. If not, load CMS by issuing IPL 190 CLEAR. Wait for the CMS version identification to appear on the screen. Then press **ENTER**.
2. Invoke the SAMPNSS EXEC:

```
sampnss cms cmsxa ■
```

The SAMPNSS EXEC issues the DEFSYS command.

Note: If you use "CMS" (the default name) for the System/370 name, users will receive the new level of CMS whenever they issue the command IPL CMS.

```
HCPNSD440I Named Saved System (NSS) CMS was
           successfully defined in fileid
           fileno

HCPNSD440I Named Saved System (NSS) CMSXA was
           successfully defined in fileid
           fileno
```

The SAMPNSS EXEC first issues the DEFSYS command to define a skeleton system data file for CMS and for CMSXA. These messages tell you the DEFSYS commands were successful.

The names shown in the SAMPNSS command are the default names for the System/370 (CMS) and for the 370-XA (CMSXA) systems. Of course, you may name these systems whatever you wish; but if you change the names you must either edit the SAMPNSS EXEC to use your names or define your systems manually.

Note: The names are positional on the command (the System/370 name followed by the 370-XA name).

```
OWNERID  FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMS      NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The SAMPNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

3. Issue the QUERY NSS ALL MAP command to make sure CMS is defined properly. Check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
:
nnnn CMS      NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Set your machine mode to 370, load CMS (IPL 190), and save the CMS system:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

If your machine is already in 370 mode, you will not get a response.

```
ipl 190 clear parm savesys cms ■
HCPNSD440I Named Saved System (NSS) CMS was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMS.

```
DMSWSP327I The installation saved segment
   could not be loaded
```

Disregard the message about not loading the saved segment; CMS has been successfully saved as a named saved system.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

5. Set your machine mode to XA, load CMS (IPL 190), and save the CMSXA system:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

If your machine is already in 370-XA mode, you will not get a response.

```
ipl 190 clear parm savesys cmsxa ■
HCPNSD440I Named Saved System (NSS) CMSXA was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMSXA.

```
DMSWSP327I The installation saved segment
   could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

Disregard the message about not loading the saved segment; CMSXA has been successfully saved as a named saved system.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

6. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the CMS (or CMSXA) named saved system. The example below shows loading CMS in System/370 mode:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
define storage 2m ■
```

If your machine is already in 370 mode, you will not get a response.

```
STORAGE = 2M
Storage cleared - system reset
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

Defining storage causes a system reset.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

After receiving the CMS version identification, press **ENTER** to complete CMS or CMSXA initialization.

7. Check the CMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Check whether the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss

query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Redefine your storage before you continue with the following steps:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 30. Install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments

If you plan to use the CMS/DOS environment or the VSAM product, you must install the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS saved segments.

If you do not want to install these saved segments, go to "Step 31. Create the CMSINST and HELP Saved Segments" on page 301.

In this step, you will:

- Install CMSDOS and CMSBAM in separate saved segments
- Create a CMSVSAM segment and install the VSAM product tape
- Create a CMSAMS segment and install Access Method Services.

This step takes approximately 1 hour.

Warning: The CMSDOS and CMSBAM segments must be installed before you install CMSVSAM and CMSAMS.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. IPL 190. This is the CMS system disk.

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
```

VM/XA CMS 5.6 *mm/dd/yy hh:mm* ** *DO NOT press ENTER!* **	Your own installation header appears if you have changed it.
`access (noprof ■`	This command suppresses execution of MAINT's PROFILE EXEC.

Ready; T=*n.nn/n.nn hh:mm:ss*
`access 193 e ■`
`access 5E5 b ■`
Ready; T=*n.nn/n.nn hh:mm:ss*
	Access the disks with installation and service EXECs, and the DOSGEN EXECs.

`set emsg on ■` Ready; T=*n.nn/n.nn hh:mm:ss*	You want to see any and all error messages during execution of the installation EXEC.

2. Define a segment into which the DOSGEN EXEC can load CMSDOS:

`defseg dosinst 900-90f sr ■` HCPNSD440I Saved segment DOSINST was successfully defined in fileid *fileno* Ready; T=*n.nn/n.nn hh:mm:ss*	This command defines a 1M segment for DOSINST starting at 900000. You may place this segment anywhere below the segment spaces defined for CMSDOS, CMSBAM, CMSVSAM, and CMSAMS.

3. Invoke the DOSGEN EXEC with a load address and the name DOSINST. The load address used for DOSINST in this example is 900000.

```
dosgen 900000 dosinst ■
HCPNSS440I Saved segment DOSINST was
            successfully saved in fileid
            fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWGN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the DOSINST segment. Error messages for the DOSGEN EXEC are listed on page 295.

4. IPL CMS:

```
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Define CMSDOS, CMSBAM, CMSAMS, and CMSVSAM:

```
set sysname cmsdos dosinst ■
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsdos ■
HCPNSD440I Saved segment CMSDOS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss DOSBAM   DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSDOS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsbam ■
HCPNSD440I Saved segment CMSBAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSBAM   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsams ■
HCPNSD440I Saved segment CMSAMS was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSAMS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
sampnss cmsvsam ■
HCPNSD440I Saved segment CMSVSAM was successfully
            defined in fileid fileno

OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSVSAM  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Invoke the DOSGEN EXEC with a load address and the name CMSDOS. The load address set up by SAMPNSS is B00000 for the name CMSDOS.

```
access 193 e ■
Ready; T=n.nn/n.nn hh:mm:ss
dosgen b00000 cmsdos ■
HCPNSS440I Saved segment CMSDOS was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSGEN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the CMSDOS segment. Error messages for the DOSGEN EXEC are listed on page 295.

7. To save the load map, rename it and copy it to the alternate LOCAL minidisk (395):

```
access 395 d ■
DMSACC724I 395 replaces D(395)
Ready; T=n.nn/n.nn hh:mm:ss
copy load map a cmsdos segmap d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Error Messages from DOSGEN

If DOSGEN detects an error in the address that you specified:

```
DMSGEN095E INVALID ADDRESS
```

If DOSGEN cannot find a read/write accessed A-disk:

```
DMSGEN006E NO READ/WRITE A-DISK ACCESSED
```

If DOSGEN finds unresolved external references while loading the text files:

```
DMSGEN111E DOSGEN FAILED DUE TO LOAD ERRORS
```

If DOSGEN detects an error while assigning the storage key or saving the segment:

```
DMSGEN412S DOSGEN FAILED DUE TO SETKEY ERRORS

DMSGEN141S DOSGEN FAILED DUE TO SAVESYS ERRORS
```

8. Replace the name CMSBAM in the CMS SYSNAME table with any name that is **not** a name previously used as a segment name:

```
set sysname cmsbam sysname ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The SET SYSNAME command enters an alternate name for CMSBAM in the SYSNAME table. Choose a name that was **not** used previously as a segment name. This command does not actually change the saved segment's name, but keeps CP from finding and IPLing the segment at the wrong time.

9. Place your CMS virtual machine in a CMS/DOS environment, then invoke SAMGEN to load the CMSBAM segment. You must give the EXEC an address at which to load the CMSBAM segment; this address also must match the address in the skeleton CMSBAM segment that you defined beforehand.

```
set dos on ■
```

This command places your virtual machine in a CMS/DOS environment.

```
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
              at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
samgen ■
DMSSGN363R ENTER LOCATION WHERE CMSBAM
WILL BE LOADED AND SAVED:
```

You may receive these informational messages. *sysname* is the name you chose in substep 8.

```
b10000 ■
DMSSGN364I FETCHING CMSBAM...
DMSFET710I Phase DMSVBM entry point at location B100C0.
DMSSGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsbam ■
HCPNSS440I Saved segment CMSBAM was successfully
             saved in fileid fileno
DMSSGN365I SYSTEM CMSBAM SAVED
Ready; T=n.nn/n.nn hh:mm:ss
```

The messages indicate that the segment has been loaded and saved.

10. Set DOS off:

```
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

11. Define and format a minidisk for VSAM files.

    a. Map your USER DIRECT and examine the map to find a space for the new minidisk. Look for a gap of appropriate size and record the starting cylinder of that gap. The number of cylinders needed depends on the DASD type:

Device	Cylinders Needed
3350	15 cylinders
3375	22 cylinders
3380	14 cylinders
3390	14 cylinders

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
diskmap user direct ■
File USER DISKMAP A has been created.
Ready; T=n.nn/n.nn hh:mm:ss
xedit user diskmap ■
locate/GAP ■
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Refer to the directory map to find a gap. Repeat the **locate** command as many times as necessary. When you find a suitable gap, note the starting cylinder and DASD label the cylinders reside on and leave the file.

b. XEDIT your directory and add an entry for the new minidisk:

```
xedit user direct ■
locate/USER MAINT ■
locate/MDISK ■
input ■
MDISK vdevno devtype startcyl cyls label MW ALL ■
■
file ■
```

Locate the MAINT minidisk definitions.

vdevno, *devtype*, and *label* are the address, device type, and label of the DASD volume where the new minidisk is located. *startcyl* is the starting cylinder of the gap where the new minidisk is defined. *cyls* is the number of cylinders needed for the minidisk.

Press **ENTER** to return to the command line.

c. Map the directory again and examine the map to make sure that the new minidisk was created in the place you specified and that it does not overlap any existing minidisks:

```
diskmap user direct ■
```

d. Issue DIRECTXA to update the directory and bring it online:

```
directxa user ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
EOJ DIRECTORY UPDATED AND ONLINE
Ready; T=n.nn/n.nn hh:mm:ss
```

e. Link to the new minidisk:

```
link * vdevno vdevno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

f. Format the new minidisk:

```
format vdevno v (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk V(vdevno).
            Do you wish to continue?
            Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
label ■
DMSFOR733I Formatting disk V
DMSFOR732I nn cylinders formatted on V(vdevno)
Ready; T=n.nn/n.nn hh:mm:ss
```

g. Access the new minidisk as A:

```
access vdevno a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

12. Set DOS on:

```
set dos on ■
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
              at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
```

You may receive these informational messages. *sysname* is the name you chose in substep 8 on page 296.

13. Mount the VSAM product tape on a tape drive attached to MAINT at virtual address 181. (If you do not have the proper class authority to attach a tape, the system operator may have to do it for you.)
14. Invoke VSAMGEN EXEC:

```
vsamgen ■
SELECT ONE OF THE FOLLOWING FUNCTIONS BY ENTERING THE NUMBER:

 1. INSTALL AMS            (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 2. INSTALL VSAM           (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 3. INSTALL VSAM AND AMS   (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 4. BUILD  AMS             (BUILD DOSLIB, CREATE SEGMENT)
 5. BUILD  VSAM            (BUILD DOSLIB, CREATE SEGMENT)
 6. BUILD  VSAM AND AMS    (BUILD DOSLIB, CREATE SEGMENT)
 7. RESTART AMS            (CREATE SEGMENT)
 8. RESTART VSAM           (CREATE SEGMENT)
 9. RESTART VSAM AND AMS   (CREATE SEGMENT)
10. QUIT                   (EXIT VSAMGEN EXECUTION)

      ENTER RESPONSE...
```

Choosing option 3 tells the EXEC to create both CMSVSAM and CMSAMS segments as new segments.

If this is the initial installation of VSAM, select function **1**, **2**, or **3**.

If the text files have already been created from the VSAM product tape and are currently on the accessed A-disk, select function **4**, **5**, or **6**.

If the DOSLIBs currently reside on an accessed disk, select function **7**, **8**, or **9**. In this case, it is not necessary to have the text files available.

```
3 ■

DMSVGN365R ONE OR MORE OF THE TEXT FILES LISTED IN THE BOTH EXEC
            ARE MISSING.  THE VSAM PP PID TAPE SHOULD BE ON TAPE DRIVE 181
            TO RESTORE THE FILES. ENTER:
            'GO' IF TAPE DRIVE IS READY TO LOAD FILES,
            'QUIT' TO STOP GENERATION PROCESS.
go ■
```

Messages from VSAMGEN

While VSAMGEN is processing, you may receive error and information messages. These messages are self-explanatory. Messages labeled 2101I are information messages from the linkage editor and may be ignored.

Respond to the EXEC messages as they appear:

```
DMSVGN362I LINK-EDITING CMSVSAM ...
DMSVGN363I CMSVSAM DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSVSAM SHARED WILL BE
         LOADED AND SAVED:

ba0000 ■

DMSVGN363R ENTER LOCATION WHERE CMSVSAM NONSHARED WILL BE
         LOADED AND SAVED:
a30000 ■

DMSVGN364I FETCHING CMSVSAM ...
DMSFET710I PHASE DMSVVS ENTRY POINT AT LOCATION nnnnnn
DSMVGN371R CMSVSAM IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:

cmsvsam ■

DMSVGN365I SYSTEM CMSVSAM SAVED.

DMSVGN368R ERASE CMSVSAM DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■

DMSVGN362I LINK-EDITING CMSAMS ...
DMSVGN363I CMSAMS DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSAMS SHARED WILL BE
         LOADED AND SAVED:
```

This name is the default name with which the segment was defined.

```
b40000 ■
DMSVGN363R ENTER LOCATION WHERE CMSAMS NONSHARED WILL BE
         LOADED AND SAVED:
a00000 ■
DMSVGN364I FETCHING CMSAMS ...
DMSFET710I PHASE DMSVAS ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAN ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAX ENTRY POINT AT LOCATION nnnnnn
DMSVGN371R CMSAMS IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsams ■
DMSVGN365I SYSTEM CMSAMS SAVED.

DMSVGN368R ERASE CMSAMS DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the recommended location.

This name is the default name used for defining the segment.

15. You may now purge the DOSINST named saved segment:

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
purge nss fileno ■
No files purged
0001 file pending purged
Ready; T=n.nn/n.nn hh:mm:ss
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file identifier for DOSINST.

Step 31. Create the CMSINST and HELP Saved Segments

In this step, you will:

- Define a segment to contain EXECs and System Product Editor macro instructions
- Use the DCSSGEN command to load and save the CMSINST segment
- Save the HELP file directory in a saved segment.

Notes:

1. You must have the NAMESAVE HELP statement in your user directory in order to save the HELP file directory information in a saved segment.

   See "Step 21. Update the User Directory" on page 71.

2. If the HELP segment is defined as a member of a segment space, you must have a NAMESAVE statement (for example, NAMESAVE INSTHELP) in the directory entry for the user who invokes SAVEFD. SAVEFD specifies the name of the segment space.

This step takes approximately 20 minutes.

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. Log on as MAINT (unless you are continuing from the previous step).
2. Check your virtual storage. If it is less than 16M, issue the following:

```
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 0016M
```

3. IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

■

Press **ENTER** to complete the CMS initialization.

```
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Use the SAMPNSS EXEC to define a segment for CMSINST:

   Note: If you have increased the number of entries in the CMS load list, be sure that the DEFSEG entry for CMSINST in the SAMPNSS EXEC has enough pages defined to accomodate the increased size.

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
           defined in fileid fileno
```

```
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The EXEC issues a QUERY command that displays information about the segments defined for named saved systems.

5. Define the HELP segment:

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
             defined in fileid fileno
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Save both segments:

```
saveseg cmsinst ■
HCPNSS440I Saved segment CMSINST was
             successfully saved in fileid
             fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg help ■
HCPNSS440I Saved segment HELP was
             successfully saved in fileid
             fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Each member saved segment defined by SAMPNSS must be saved separately.

7. Redefine each segment to create the skeleton segments:

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
             defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If you do not want to use the sample load list, CMSINST EXECLIST, for the DCSSGEN command, use the System Product Editor (XEDIT) to create your own load list:

```
xedit instlist file a ■
input ■

* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S  ■

■
file ■
```

This is a sample. Enter the load list entries you need.

Only one comment line per list is allowed; it must be the first line. For more information on creating this list, see "DCSSGEN Command" on page 580.

The remainder of the lines contain entries for frequently used EXECs and Editor macros; multiple users can share the same executing copy of these EXECs and macros.

Press **ENTER** to return to the command line.

9. The load list must have a fixed record length in order for you to use the DCSSGEN command. If your load list has variable length records, change it to a fixed length file by issuing:

```
copyfile fn ft fm = = = (recfm f lrecl 80 replace ■
```

10. Invoke the DCSSGEN command:

```
access 193 e ■
dcssgen fn ft fm cmsinst ■
```

fn ft fm is the fileid of the file that contains the list of EXECs and Editor macros to be loaded into the segment, either CMSINST EXECLIST *fm* or your own file (for example, INSTLIST FILE A).

CMSINST is the name of the segment you want to create. It is also the default name of the installation saved segment in the DMSNGP profile. If you changed the name in the DMSNGP profile, or specified a different name in response to the installation questions (see page 248), use that name instead. If you do not specify a segment name, the DCSSGEN command defaults to CMSINST.

```
HCPNSS440I Saved segment CMSINST was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

The DCSSGEN command creates a load map file, CMSINST DCSSMAP A. For the sample load list created in substep 8 on page 303, the load map would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
    15:41:59 10/22/85 copy of RDRLIST  EXEC     S loaded as MAIL     EXEC
    EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
    15:41:58 10/22/85 copy of FILELIST EXEC     S loaded as FILELIST EXEC
    EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
     7:30:18 11/26/85 copy of SYSPROF  EXEC     S loaded as SYSPROF  EXEC
    EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
     8:47:55 12/18/84 copy of PARSE    XEDIT    S loaded as PARSE    XEDIT
    EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
    15:40:32 10/22/85 copy of DISCARD  EXEC     S loaded as DISCARD  EXEC
    EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
    11:03:53 10/24/85 copy of NOTE     EXEC     S loaded as NOTE     EXEC
    EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
    15:41:55 10/22/85 copy of PROFNOTE XEDIT    S loaded as PROFNOTE XEDIT
    EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
    15:41:05 10/22/85 copy of ALL      XEDIT    S loaded as ALL      XEDIT
    EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
    *** End of Source List ***
    CMSINST built at 15:56:34 on 12/02/85
```

Note: When you built your CMS nucleus, if you indicated in the DMSNGP file (USEINST=YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs CMS. If you previously indicated that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild CMS. (To find the latest version of the DMSNGP file, issue SETUP to access all the minidisks where it might be; then issue FILELIST DMSNGP ASSEMBLE *. All the copies of DMSNGP ASSEMBLE will be listed, with the date when each one was last modified.)

Messages from DCSSGEN

While DCSSGEN is processing, you may receive error or warning messages that indicate specific conditions. If errors were encountered, after processing is complete you receive the following prompt:

```
DMSEXG298R An error has been detected while building the DCSS.
           Do you still want the DCSS saved?
           Enter 1 (YES) or 0 (NO).
```

Enter **1** to disregard the error(s) and save the segment, or enter **0** to not save the segment. If you do not save the segment, you receive the message:

DMSEXG288I *segname* not saved

If DCSSGEN encounters an error while saving the segment, you receive the message:

DMSEXG288E *dcssname* not saved

If your virtual machine is not large enough to contain the segment (you need 16M), you receive the message:

```
DMSEXG284E The DCSS is not completely inside the virtual machine
```

To correct this situation, increase the size of your virtual machine, re-IPL CMS, and reissue the DCSSGEN command.

11. Define your virtual storage **less** than the address at which the HELP segment is to be loaded. For example, if the HELP segment is defined at X'C50000' (the default address in the SAMPNSS EXEC) define your storage as 12M.

```
define storage 12m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 12M
```

12. Re-IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

13. Issue the following commands to initialize and save the segment:

    (For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*.)

```
savefd init vaddr label help ■
Ready; T=n.nn/n.nn hh:mm:ss
savefd save vaddr label help ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is 19D for mixed-case American English, 19C for uppercase American English.

label is the CMS label assigned to the disk. (The labels in the sample directory are MNT19D and MNT19C.)

14. Verify that the CMSINST and CMSHELP segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss INSTHELP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss CMSINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss HELP        NSS      MAINT
```

The system data file should now have class A (rather than S) and have one user (MAINT).

15. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 32. Back Up the Named Saved System Files

In this step, you will back up all the named saved systems, including CMS, on tape.

This step takes approximately 10 minutes.

1. Mount a scratch tape with ring on an available 3420, 3422, or 3430 tape drive, or insert a cartridge with the tab protector set to off in an available 3480 tape unit. Follow machine operations manuals to mount the tape.
   - **Do not** attach the tape unit to MAINT.
   - The tape must be suitable for a 3420, 3422, 3430, or 3480 tape unit.
2. Issue the SPTAPE command to dump the CMS system data file to tape:

```
sptape dump vdevno nss all run ■
```

Substitute the virtual device number of the tape drive for the value *vdevno*. The operand RUN specifies that the SPTAPE rewinds and unloads the tape after the operation.

```
DUMPING vdevno NSS *NSS      nnnn A
⋮
SPTAPE DUMP FUNCTION ON DRIVE vdevno
   COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The messages from SPTAPE tell you that the file is being dumped to tape. The CMS ready message may occur between the messages.

3. Store the tape for emergency use. If it is ever necessary, you can use this tape and the SPTAPE command to restore the CMS system data file. For more information about the SPTAPE command, see *VM/XA SP Real System Operation*.

To restore your CMS named saved system:

1. Log on as MAINT.
2. Mount the backup tape on a tape drive.
3. Issue:

```
sptape load vdevno nss all run ■
LOADING vdevno NSS *NSS nnnn A    NOW nnnn
⋮
LOADING vdevno NSS *NSS nnnn A    NOW nnnn
SPTAPE LOAD FUNCTION ON DRIVE vdevno COMPLETE
```

4. IPL CMS:

```
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 33. Store a Backup Copy of CP on Tape

In this step, you will use DDRXA to store a backup copy of CP on tape.

The time to perform this step varies based on the device and contention for the device's channel. It may take up to 30 minutes.

1. Mount a scratch tape (with ring) on a tape drive. Follow machine operations manuals to mount the tape.
2. Attach the tape drive to MAINT at virtual device number 0181:

```
attach rdevno * 181 ■
TAPE     0181 ATTACHED
Ready; T=n.nn/n.nn hh:mm:ss
```

The ATTACH command attaches the real device (*rdevno*) to MAINT's virtual machine at virtual device number 0181.

3. Load the DDRXA utility to tape. If your tape drive can operate at a density of 6250 BPI, enter the commands under 3a. Otherwise, enter the commands under 3b.
   a. If your tape drive can operate at a density of 6250 BPI, enter these commands:

```
tape modeset (tapl den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Set the tape density to 6250 BPI. If you receive message DMSPAR622E, reissue the TAPE MODESET command.

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl (den 6250 ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

   Go to substep 4.

   b. If your tape drive cannot operate at a density of 6250 BPI, enter these commands:

```
filedef in disk ipl ddrxa s ■
Ready; T=n.nn/n.nn hh:mm:ss
filedef out tapl ■
Ready; T=n.nn/n.nn hh:mm:ss
movefile in out ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Because this step uses the MOVEFILE command, you must first use the FILEDEF command to establish data definitions for files to be copied.

4. Change MAINT's virtual processor configuration to 370-XA architecture:

```
set machine xa ■
System reset
System = XA
```

Setting the virtual machine to 370-XA architecture causes a reset as if you entered SYSTEM CLEAR. If your machine is already in 370-XA mode, you will not get a response.

```
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Rewind the scratch tape on virtual device number 0181:

```
rewind 181 ■
Rewind complete
```

6. Load (IPL) the tape and answer the prompts from DDRXA:

```
ipl 181 clear ■
```

Wait a few moments for DDRXA to prompt you. If a prompt does not appear, press the **ENTER** key.

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:
```

```
sysprint cons ■
ENTER:
```

This first control statement tells DDRXA that you want program messages sent to your console.

```
input vdevno devtype xasres ■
ENTER:
```

The second control statement is the input control statement. *vdevno* is the virtual address of the system residence volume. (The virtual address in the sample HCPSYS ASSEMBLE file and USER DIRECT is 123.) *devtype* is the DASD type of this volume.

```
output 181 devtype (mode 6250 compact ■
ENTER:
```

This control statement specifies the device to which you are dumping the system.

If your tape drive is not operating at 6250 BPI density, omit the **mode 6250 compact** option on this command.

```
dump cpvol ■
```

The statement DUMP CPVOL dumps cylinder 0 and all PERM and DRCT DASD space from the system residence volume to the tape.

```
DUMPING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn    0000   nnnn
END OF DUMP
BYTES IN nnnnnnnnnn BYTES OUT nnnnnnnnnn
TRACKS NOT COMPACTED ON TAPE - nnnnnnnnnn
```

These are informational messages. GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

```
ENTER:
■
END OF JOB
```

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

You now have a loadable tape of the system. If you need to use this backup copy to restore CP, perform these steps **on a first-level system**:

1. Mount the backup tape on a tape drive.
2. Set your virtual machine to 370-XA architecture, define your virtual storage as 16 megabytes, and IPL CMSXA:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
define storage 16m ■

STORAGE = 16M
Storage cleared - system reset
ipl cmsxa ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

If your machine is already in 370-XA mode, you will not get a response.

3. Perform a hardware load (IPL) of the tape device.
4. Use DDRXA to restore the system to disk:

```
VM/XA SYSTEM PRODUCT DASD DUMP/RESTORE PROGRAM
ENTER CARD READER ADDRESS OR CONTROL STATEMENTS
ENTER:

sysprint cons ■
ENTER:

input rdevno devtype ■
ENTER:

output rdevno devtype xasres ■
ENTER:

restore all ■

RESTORING XASRES
DATA DUMPED mm/dd/yy
  AT hh.mm.ss GMT FROM XASRES
  RESTORED TO XASRES
INPUT CYLINDER EXTENTS OUTPUT CYLINDER EXTENTS
      START    STOP    START  STOP
      0000     nnnn     0000   nnnn
⋮
END OF RESTORE
BYTES RESTORED nnnnnnnnnn

ENTER:
■
END OF JOB
```

This first control statement tells DDRXA that you want program messages sent to your console.

The second control statement is the input control statement. Identify the device number (*rdevno*) and tape device type (3420, 3422, 3430, or 3480) where the backup tape is mounted.

This control statement specifies the DASD device to which you are restoring the system (XASRES).

The RESTORE ALL statement tells DDRXA to restore the whole tape to the output device.

Informational messages: GMT means Greenwich Mean Time.

The exact cylinder extents vary according to the device type.

When DDRXA finishes, it prompts you with ENTER. To end the program, press the **ENTER** key.

Chapter 5. Installing a New System National Language

Overview

VM/XA System Product Release 2.1 is shipped with mixed-case American English as the system default national language. During installation, you can change the system default to uppercase American English. When VM/XA System Product Release 2.1 is completely installed, the system national language is automatically set for CP and all virtual machines on the system.

Note: A user's virtual machine is set to the default system national language during logon, unless the directory entry for that user ID contains an OPTION LANG statement that overrides the system national language. (See *VM/XA SP Planning and Administration.*)

You can install other national languages on your system:

- To replace the current default system national language. Use the procedure described in this chapter.
- As an option available to selected system users. See *VM/XA SP Planning and Administration.*

National languages (except mixed-case American English and uppercase American English) are distributed on national language feature tapes: one VM/XA System Product Release 2.1 tape for each language. The files on a national language feature tape contain translated information. The national language is identified by a unique 1-to 5-character *langid*, shown in Table 5.

Table 5. National Language *langids* and Country Codes

Language	Langid	Country Code (*y*)
Mixed-case American English	AMENG	None
Kanji (Japan)	KANJI	A
Uppercase American English	UCENG	B
Brazilian Portuguese	PORTG	C
French	FRANC	D
German	GER	E

To install a language as the new default system national language:

1. Load the language files from the national language feature tapes to minidisks.
2. Identify the *langid* of the new default system national language in the system generation profiles and update the CMS translation tables (if necessary).
3. Build a new CMS nucleus containing the CMS language files.
4. Create the CMS and CMSXA named saved systems.
5. Create the CMSINST and HELP segments.
6. Build a new CP nucleus containing the CP language files.
7. Shut down the system and do a warm start to bring the new CP and CMS nuclei on line.
8. **If GCS is installed,** create a new GCS configuration file (optional).
9. **If GCS is installed,** build a new GCS nucleus containing the GCS language files.

Contents of the National Language Feature Tapes

A national language feature tape for VM/XA System Product Release 2.1 contains two types of files:

- Source files
- Object code files.

Figure 5 shows the layout of these files on the tape.

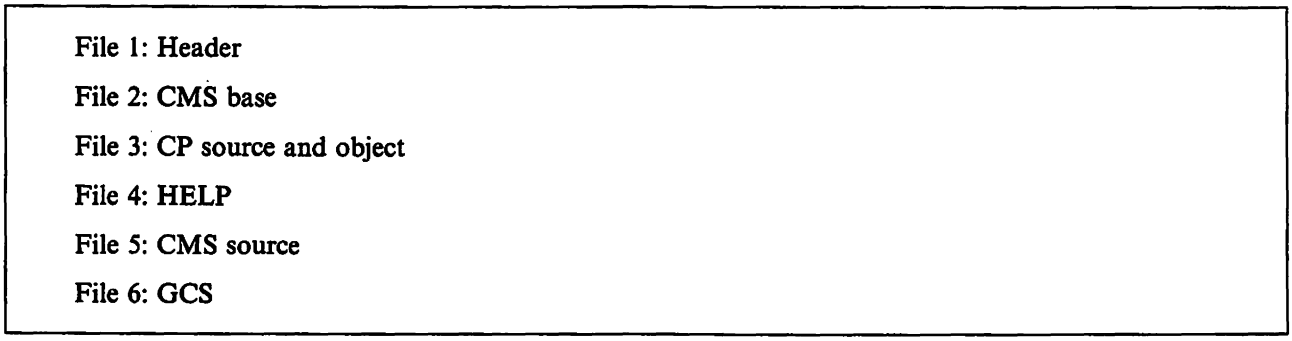


Figure 5. National Language Feature Tape Layout

Source Files

The national language source files can be edited. These source files must then converted into object files before the system can use them.

Table 6. National Language Source Files

File ID		Description
HCPMES[*y*]	REPOS	CP message repository, containing the translated versions of CP system messages.
DMSMES[*y*]	REPOS	CMS message repository, containing the translated versions of CMS system messages.
DMSSPA[*y*]	DLCS	CMS command syntax file, containing the translated syntax definitions for CMS commands.
DMSTRT[*y*]	ASSEMBLE	CMS uppercase translate table, mapping lowercase alphabetic characters to uppercase for the language.
CSIMES[*y*]	REPOS	GCS message repository, containing the translated versions of GCS messages.
xxxxxxxx	HELP*cccc*	HELP files, where *xxxxxxxx* is the command name and *cccc* is the name of the HELP facility component. HELP files have the same file IDs, regardless of the language. The HELP files for each language installed on your system must therefore reside on a different disk.

y is a "country code" character that identifies the national language and *langid*. The recognized values for this country code character are shown in Table 5 on page 311. These values are stored in a file called VMFNLS LANGLIST.

Note: The mixed-case American English versions of translatable source files have 6-character filenames; they do not use the country code character.

Object Files

National language object files are source files that have been converted to executable form. These files are used to build the new CP, CMS, or GCS nucleus during the installation procedure.

Table 7. National Language Object Files

File ID		Description
HCPMES[*y*]	TEXT or TXT*nnnnn*	CP message repository
DMSMES[*y*]	TEXT or TXT*nnnnn*	CMS message repository
DMSSPA[*y*]	TEXT or TXT*nnnnn*	CMS system command syntax definition file
DMSSSY[*y*]	TEXT or TXT*nnnnn*	CMS system national language translation and synonym table
DMSTRT[*y*]	TEXT or TXT*nnnnn*	CMS uppercase translation table
CSIMES[*y*]	TEXT or TXT*nnnnn*	GCS message repository

y is a "country code" character that identifies the national language and *langid*. The recognized values for this country code character are shown in Table 5 on page 311. These values are stored in a file called VMFNLS LANGLIST.

The filetype of the national language object files on the national language feature tape is TEXT. When a file has been serviced, the filetype is changed to TXT*nnnnn*, where *nnnnn* is the number of the most recently applied PTF.

Note: The mixed-case American English versions of object files have 6-character filenames; they do not use the country code character.

Procedure

- To replace the current default system national language, use the procedure described in this chapter.
- To install a new national language as an option available to selected system users, see *VM/XA SP Planning and Administration.*

Notes:

1. Perform this procedure while logged on to the MAINT user ID.
2. Some languages have character sets that require special hardware. Be sure that all display devices in your configuration can properly display the character set of the new default system national language.

Step 1. Load the Language Files from Tape to Disk

1. Define and format a minidisk for the national language HELP files.

   a. Map your USER DIRECT and examine the map to find a space for the new minidisk. Look for a gap of appropriate size and record the starting cylinder of that gap. The number of cylinders needed depends on the DASD type:

Device	Cylinders Needed
3350	50 cylinders
3375	63 cylinders
3380	40 cylinders
3390	38 cylinders

   You should also define a service disk for your new default system national language at this time. It should be the same size as the new HELP file disk. If the gap where you define your HELP file disk is big enough, you can define the service disk next to the HELP file disk; if not, find another gap.

```
access 295 d ■
Ready; T=n.nn/n.nn hh:mm:ss
diskmap user direct ■
File USER DISKMAP A has been created.
Ready; T=n.nn/n.nn hh:mm:ss
xedit user diskmap ■
locate/GAP ■
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Refer to the directory map to find a gap. Repeat the **locate** command as many times as necessary. When you find a suitable gap, note the starting cylinder DASD label the cylinders reside on and and leave the file.

   b. XEDIT your directory and add an entry for the new minidisk:

```
xedit user direct ■
locate/USER MAINT ■
locate/MDISK ■
input ■
MDISK 19B devtype startcyl cyls label MW ALL ■
MDISK 49B devtype startcyl cyls label MW ALL ■
```

Locate the MAINT minidisk definitions.

This example uses MAINT 19B as the new minidisk and MAINT 49B as the new service minidisk. *devtype* and *label* are the device type and label of the DASD volume where the new minidisk is located. *startcyl* is the starting cylinder of the gap where the new minidisk is defined. *cyls* is the number of cylinders needed for the minidisk.

■ Press **ENTER** to return to the command line.
file ■

c. Map the directory again and examine the map to make sure that the new minidisks were created in the place you specified and that they do not overlap any existing minidisks.

d. Issue DIRECTXA to update the directory and bring it online:

```
directxa user ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.1
HCPDIR750W RESTRICTED PASSWORD FILE NOT FOUND
EOJ DIRECTORY UPDATED AND ONLINE
Ready; T=n.nn/n.nn hh:mm:ss
```

e. Link to the new minidisks:

```
link * 19b 19b ■
Ready; T=n.nn/n.nn hh:mm:ss
link * 49b 49b ■
Ready; T=n.nn/n.nn hh:mm:ss
```

f. Format the new minidisks:

```
format 19b v (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk V(19B).
             Do you wish to continue?
             Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
mnt19b ■
DMSFOR733I Formatting disk V
DMSFOR732I nn cylinders formatted on V(19B)
Ready; T=n.nn/n.nn hh:mm:ss
format 49b v (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk V(49B).
             Do you wish to continue?
             Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
mnt49b ■
DMSFOR733I Formatting disk V
DMSFOR732I nn cylinders formatted on V(49B)
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Attach a tape drive as to MAINT as 181:

```
attach vdevno * 181
Ready; T=n.nn/n.nn hh:mm:ss
```

3. Mount the national language feature tape on device 181. The internal tape label (not the label pasted on the reel or cartridge) must match the label specified in SPLOAD PROFILE. The label in the SPLOAD PROFILE supplied by IBM is R2M1NLS. Since the label varies from tape to tape, this is almost

certainly **not** the same as the internal tape label. To find out what the internal tape label is, load the first tape file from the feature tape and XEDIT the tape header:

```
vmfplc2 load ■
LOADING...
 $TAPE HEADER A3
 END-OF-FILE OR END OF TAPE
Ready; T=n.nn/n.nn hh:mm:ss
xedit $tape$ header ■
TAPENO = 1 ; TAPEFORMAT = label
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Change every occurrence of the label in the SPLOAD PROFILE to match the internal tape label:

```
xedit sploadprofile ■
change/R2M1NLS/label/* * ■
file ■
```

4. Invoke ITASK EXEC with the LOAD LANG *filegroup* parameters:

   Warning: The SPLOAD PROFILE determines the minidisk to which each tape file will be loaded from the product tape. If you change the the minidisk addresses in the SPLOAD PROFILE, you must also change the ITASK EXEC and the product parameter file to agree with it.

```
itask load lang all ■
```

Enter **all** to load all of the files. Enter **allobj** instead to load only the TEXT, REPOS, and HELP*cccc* files. If you do not have all the VM/XA System Product Release 2.1 components installed in your system, issue the command with the **cp**, **cms**, **gcs**, **help**, or **cmssrc** operand to load the files for each component that you do have installed. See "ITASK EXEC" on page 611.

```
DMSWSL409I Loading CP OBJECT to the 194 disk
           attached to MAINT
DMSWSL409I Loading CMS BASE to the 193 disk
           attached to MAINT
DMSWSL409I Loading GCS OBJECT to the 595 disk
           attached to MAINT
DMSWSL737R Enter the minidisk address for
           the HELP files:
19b ■
DMSWSL409I Loading HELP FILES to the 19B disk
           attached to MAINT
DMSWSL409I Loading CMS SOURCE to the 393 disk
           attached to MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

ITASK invokes SPLOAD EXEC to load the language files to the destinations specified in SPLOAD PROFILE, as indicated in Table 8 on page 317 and Table 9 on page 317.

Notes:

1. REPOS files for source are loaded to the object disks for the corresponding components.
2. You are prompted for the address of the disk where you want the HELP files loaded.

Table 8. National Language Source File Minidisk Destinations	
Source File	**Minidisk**
HCPMES*y* REPOS	MAINT 194
DMSMES*y* REPOS	MAINT 193
DMSSPA*y* DLCS	MAINT 393
DMSTRT*y* ASSEMBLE	MAINT 393
CSIMES*y* REPOS	MAINT 595
xxxxxxxx HELP*cccc*	MAINT *vdev* (unique HELP disk for this language). The recommended address for the default system national language is 19B.

Table 9. National Language Object File Minidisk Destinations	
Object File	**Minidisk**
HCPMES*y* TEXT	MAINT 194
DMSMES*y* TEXT	MAINT 193
DMSSPA*y* TEXT	MAINT 193
DMSSSY*y* TEXT	MAINT 193
DMSTRT*y* TEXT	MAINT 193
CSIMES*y* TEXT	MAINT 595

Step 2. Specify the Langid of the New Language in the System Generation Profiles and Update the CMS Translation Tables

1. Establish the correct minidisk access order:

```
setup ■
⋮
```

2. Modify the product parameter file:

```
xedit 56643089 $ppf ■
⋮
file ■
```

The :NLS. statements in the :CP., :CMS., and :GCS. component areas of the product parameter file identify the *langid* of the system national language for each nucleus. (There is no national language support for the dump viewing facility.) Change these statements to specify the *langid* of the language that you are installing. **Make sure that you specify the SAME language in each statement.**

3. Modify the CMS nucleus generation profile (DMSNGP ASSEMBLE):

```
xedit dmsngp assemble ■
```

```
locate/HELP ■
change/vaddr/19B/ ■
```

Change the HELP disk address to 19B.

```
locate/LANGID ■
change/AMENG/langid/ ■
file ■
```

Change the LANGID statement to identify the new language.

If the new language contains Double-Byte Character Set data, change the DBCS statement to YES.

4. Assemble the modified DMSNGP file:

```
itask assemble dmsngp ■
⋮
DMSUPD181E No update files were found
DMSWHM1907I Assembling DMSNGP
DMSWHM1909I DMSNGP TEXT A created
PRT FILE fileno SENT FROM MAINT   PUN AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

ITASK invokes the SETUP EXEC to reestablish the minidisk access order.

5. Copy the text deck from MAINT 191 to MAINT 395, then erase it from MAINT 191:

```
copy dmsngp text a = = b (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase dmsngp text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The DMSTRT*y* ASSEMBLE file that you loaded from the language feature tape in Step 1 contains read-only system translation tables used by other CMS modules for:

- Uppercase translation
- 327x workstation support (displayable characters, APL, TEXT, and so on).

You may need to update the translation tables in this file to match the language customization generated for your 3174/3274 controllers. You should create a language DCSS or CMS named saved system for each national language that you install that requires customization of your 3174/3274 controllers.

For example, you can edit DMSTRT*y* using the XEDIT command with the UPDATE option:

```
setup ■
xedit dmstrty assemble * (update ■
```

This option creates a file named DMSTRT*y* UPDATE that contains your changes; the source file is not changed. For more information about using the XEDIT command with the UPDATE option, refer to the *VM/XA SP System Product Editor Command and Macro Reference*.

Next, create an auxiliary control file that points to your UPDATE file, and then identify the auxiliary control file in the CMS control file (DMSXA). For more information about control files and auxiliary control files, see Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361.

Finally, use the VMFNLS EXEC to generate the associated object file (DMSTRT*y* TEXT) that includes your changes:

```
setup ■
vmfnls dmstrty assemble dmsxa ■
```

For more information about the VMFNLS EXEC, see page 672.

Step 3. Build a New CMS Nucleus Containing the CMS Language Files

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Invoke ITASK BUILD CMS:

```
itask build cms ■
   No files changed
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

ITASK invokes VMFBLD EXEC to build the nucleus, using the loadlist, control file, and *langid* specified in the product parameter file.

```
RDR FILE nnnn SENT FROM MAINT PUN AS nnnn
  RECS 026K COPY  001 N NOHOLD NOKEEP
```

The CMS nucleus has been sent to your reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

From now on, you will receive CMS messages in the new default language, even though you have not finished generating CMS.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    fileno * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 8 on page 321 below.

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

CMS Nucleus Generation Prompts and Responses

Note: Each of the following prompts appears **only** if the corresponding statement in DMSNGP is missing or empty (and the DEFNUC macro contains no default value), or if the DMSNGP statement contains a question mark (?):

```
DMSINQ606R System disk address =
```

■ — 190 is the default.

```
DMSINQ615R Y-disk address =
```

■ — 19E is the default.

```
DMSINQ640R HELP disk address =
```

■ — 19D is the default.

```
DMSINQ764R Language id =
```

■ or *langid* ■ — This response identifies the *langid* of your system national language. Text decks for this language are loaded into the CMS nucleus. The original default was **AMENG** (mixed-case American English); but you modified it in Step 2.

```
DMSINQ293R Is this a DBCS language?  Enter 1 (YES) or 0 (NO).
```

■ (or 0 ■) or 1 ■ — The default is 0 (NO). Enter **1** if the language that you are installing contains DBCS (Double-Byte Character Set) data. The only DBCS language currently supported is Kanji.

```
DMSINQ295R Language level id =
```

■ — The system national language does not use a level ID.

```
DMSINQ296R Should the installation segment be used?  Enter 1 (YES) or 0 (NO).
```

The installation segment is an optional shared saved segment, into which you can place frequently used EXECs and System Product Editor (XEDIT) macros. You install the segment **after** you install your base system, but you must indicate now whether or not you are going to use it.

■ (or 1 ■) or 0 ■

The default is 1 (YES). Enter **0** if you do not want to use the segment.

DMSINQ310R Installation segment name =

This prompt appears only if you accepted the default (or entered **1**) at the previous prompt.

■ or *segname* ■

Enter a 1-to 8-alphanumeric-character name for the installation segment, or press **ENTER** to accept the default name, CMSINST.

DMSINI729R Do you want to save the system? Enter 1 (YES) or 0 (NO).

0 ■

Answer **0** (NO). You will save CMS in Step 4.

The default is 1 (YES). **Warning:** The default response to this prompt does not agree with the sample DMSNGP file, where SAVESYS=NO.

DMSINI730R Saved system name =

You will not see this prompt if you answered **0** to the preceding prompt.

■ or *sysname* ■

The default system names are CMS and CMSXA.

DMSINI607R Rewrite the nucleus? Enter 1 (YES) or 0 (NO).

1 ■

Enter **1** to write the CMS nucleus on the disk that you specify in your response to the next prompt.

DMSINI608R IPL device address =

■

The default is the address of the system disk (190).

DMSINI609R Nucleus (CYL or BLK) address =

nnn ■

nnn is the location on the 190 system disk where the new CMS nucleus is written. Enter the correct cylinder address for your DASD type:

Device	Cylinder Address
3350	068
3375	105
3380	066
3390	062

DMSINI610R Also IPL CYL/BLK 0? Enter 1 (YES) or 0 (NO).

1 ■

Enter **1** to write the initial IPL text on cylinder/block 0 of the disk where the CMS nucleus is written.

The initial IPL text is a bootstrap program that reads the CMS nucleus from the cylinder/block where the nucleus is written (as defined in your response to prompt DMSINI609R). The initial IPL text is always written on the same cylinder/block as the nucleus. If the initial IPL text is not also written on cylinder/block 0, you must specify the cylinder/block address of the nucleus when you issue IPL commands for this system. For more information, refer to the description of the IPL command in the *VM/XA SP CP Command Reference*.

```
DMSINI611R Enter version identification:
```

The version identification is displayed each time that you IPL the CMS system you are now generating.

■ or *version* ■

You can enter up to 32 descriptive characters to identify this version and level of CMS, or you can press **ENTER** to accept the default version identification, VM/XA CMS 5.6 *mm/dd/yy hh:mm*.

```
DMSINQ612R Enter installation heading:
```

The installation heading appears at the beginning of each output file created using this CMS nucleus.

■ or *heading* ■

You can enter up to 64 descriptive characters to serve as an installation heading, or you can press **ENTER** to accept the default heading, VM/XA CONVERSATIONAL MONITOR SYSTEM.

End of CMS Nucleus Generation Prompts and Responses

	From this point on, messages and prompts will appear in the national language you are installing.	

```
DMSSLG283E The langid saved segment could not be
           found; return code 44 from SEGMENT.
```

langid is your former default system national language.

```
DMSEXG327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

This is the default version identification. If you defined your own version identification, it appears here and each time that you IPL 190 or IPL CMS.

```
⋮
SYNONYM SYN
TERMINAL MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Saving and Printing the CMS Load Map

9. The CMS load map has been spooled to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
spool prt * nohold ■
close prt ■
```

```
PRT FILE filenol SENT TO MAINT RDR AS fileno2
  RECS nnnn COPY 001 N NOHOLD NOKEEP
```

The CMS load map is spooled from your virtual printer to your virtual reader.

```
access 395 d ■
query rdr ■
```

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME TYPE DIST    USERFORM OPERFORM KEEP MSG
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss          SYSPROG STANDARD STANDARD OFF  OFF
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,100 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

```
receive fileno2 fn ft d ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print fn ft d ■
```

End of Saving and Printing the CMS Load Map

Step 4. Recreate the CMS and CMSXA Named Saved Systems

Note: For general information about CMS as a named saved system, see *VM/XA SP Guide to Saved Segments*.

1. Make sure:
   - You are logged on as MAINT.
   - You have finished loading all files to the 190 and 19E minidisks. After this step, any change to files on these disks requires resaving CMS.
   - You have 16 megabytes of virtual storage. To determine your storage size, issue QUERY VIRTUAL STORAGE. If you have less than 16 megabytes, issue DEFINE STORAGE 16M.
   - CMS is running. If not, load CMS by issuing IPL 190 CLEAR. Wait for the CMS version identification to appear on the screen. Then press **ENTER**.
2. Establish the correct minidisk order:

```
access 5E5 b ■
vmfsetup 56643089 cms ■
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

All your minidisk accesses are reestablished.

3. Invoke the SAMPNSS EXEC:

```
sampnss cms cmsxa ■
```

The SAMPNSS EXEC issues the DEFSYS command.

Note: If you use "CMS" (the default name) for the System/370 name, users will receive the new level of CMS whenever they issue the command IPL CMS.

```
HCPNSD440I Named Saved System (NSS) CMS was
      successfully defined in fileid fileno

HCPNSD440I Named Saved System (NSS) CMSXA was
      successfully defined in fileid fileno
```

The SAMPNSS EXEC first issues the DEFSYS command to define a skeleton system data file for CMS and for CMSXA. These messages tell you the DEFSYS commands were successful.

The names shown in the SAMPNSS command are the default names for the System/370 (CMS) and for the 370-XA (CMSXA) systems. Of course, you may name these systems whatever you wish; but if you change the names you must either edit the SAMPNSS EXEC to use your names or define your systems manually.

Note: The names are positional on the command (the System/370 name followed by the 370-XA name).

```
OWNERID  FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMS      NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The SAMPNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

4. Issue the QUERY NSS ALL MAP command to make sure CMS is defined properly. Check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
nnnn CMS      NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Set your machine mode to 370, load CMS (IPL 190), and save the CMS system:

```
set machine 370 ■
SYSTEM RESET
SYSTEM = 370
```

If your machine is already in 370 mode, you will not get a response.

```
ipl 190 clear parm savesys cms ■
HCPNSD440I Named Saved System (NSS) CMS was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMS.

```
DMSWSP327I The installation saved segment
   could not be loaded
```

Disregard the message about not loading the saved segment; CMS has been successfully saved as a named saved system.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

6. Set your machine mode to XA, load CMS (IPL 190), and save the CMSXA system:

```
set machine xa ■
SYSTEM RESET
SYSTEM = XA
```

If your machine is already in 370-XA mode, you will not get a response.

```
ipl 190 clear parm savesys cmsxa ■
HCPNSD440I Named Saved System (NSS) CMSXA was
            successfully saved in fileid fileno
```

Load 190 with the option to save the system under the name CMSXA.

```
DMSWSP327I The installation saved segment
   could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

Disregard the message about not loading the saved segment; CMSXA has been successfully saved as a named saved system.

```
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMS.

7. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the CMS (or CMSXA) named saved system. The example below shows loading CMS in System/370 mode:

```
set machine 370 ■                          If your machine is already in 370 mode, you will
SYSTEM RESET                               not get a response.
SYSTEM = 370
define storage 2m ■

STORAGE = 2M                               Defining storage causes a system reset.
Storage cleared - system reset
ipl cms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm

■                                          After receiving the CMS version identification,
SYNONYM SYN                                press ENTER to complete CMS or CMSXA
CP TERM MODE VM                            initialization.
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Check the CMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Check whether the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, check whether the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
Ready; T=n.nn/n.nn hh:mm:ss
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
nnnn CMS      NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Redefine your storage before you continue with the following steps:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl 190 clear ■
DMSINS327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 5. Recreate the CMSINST and HELP Saved Segments

Note: For more information about saved segments, see *VM/XA SP Guide to Saved Segments.*

1. Log on as MAINT (unless you are continuing from the previous step).
2. Check your virtual storage. If it is less than 16M, issue the following:

```
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 0016M
```

3. IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
```

If you have changed the version heading, your own heading will appear.

```
■
```

Press **ENTER** to complete the CMS initialization.

```
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Use the SAMPNSS EXEC to define a segment for CMSINST:

   Note: If you have increased the number of entries in the CMS load list, be sure that the DEFSEG entry for CMSINST in the SAMPNSS EXEC has enough pages defined to accomodate the increased size.

```
access 5E5 b ■
vmfsetup 56643089 cms ■
⋮
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

The EXEC issues a QUERY command that displays information about the segments defined for named saved systems.

5. Define the HELP segment:

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno
```

```
OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
:
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm INSTHELP DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

6. Save both segments:

```
saveseg cmsinst ■
HCPNSS440I Saved segment CMSINST was
            successfully saved in fileid
            fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg help ■
HCPNSS440I Saved segment HELP was
            successfully saved in fileid
            fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Each member saved segment defined by SAMPNSS must be saved separately.

7. Redefine each segment to create the skeleton segments:

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
:
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno

OWNERID FILETYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
:
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. If you do not want to use the sample load list, CMSINST EXECLIST, for the DCSSGEN command, use the System Product Editor (XEDIT) to create your own load list:

```
xedit instlist file a ■
input ■
```

```
* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S ■


■
file ■
```

This is a sample. Enter the load list entries you need.

Only one comment line per list is allowed; it must be the first line. For more information on creating this list, see "DCSSGEN Command" on page 580.

The remainder of the lines contain entries for frequently used EXECs and Editor macros; multiple users can share the same executing copy of these EXECs and macros.

Press **ENTER** to return to the command line.

9. The load list must have a fixed record length in order for you to use the DCSSGEN command. If your load list has variable length records, change it to a fixed length file by issuing:

   ```
   copyfile fn ft fm = = = (recfm f lrecl 80 replace ■
   ```

10. Invoke the DCSSGEN command:

    ```
    dcssgen fn ft fm cmsinst ■
    ```

    fn ft fm is the fileid of the file that contains the list of EXECs and Editor macros to be loaded into the segment, either CMSINST EXECLIST *fm* or your own file (for example, INSTLIST FILE A).

    CMSINST is the name of the segment you want to create. It is also the default name of the installation saved segment in the DMSNGP profile. If you changed the name in the DMSNGP profile, or specified a different name in response to the installation questions (see "Step 3. Build a New CMS Nucleus Containing the CMS Language Files" on page 320), use that name instead. If you do not specify a segment name, the DCSSGEN command defaults to CMSINST.

    ```
    HCPNSS440I Saved segment CMSINST was successfully
               saved in fileid fileno
    Ready; T=n.nn/n.nn hh:mm:ss
    ```

The DCSSGEN command creates a load map file, CMSINST DCSSMAP A. For the sample load list created in substep 8 on page 329, the load map would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
    15:41:59 10/22/85 copy of RDRLIST  EXEC     S loaded as MAIL     EXEC
    EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
    15:41:58 10/22/85 copy of FILELIST EXEC     S loaded as FILELIST EXEC
    EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
     7:30:18 11/26/85 copy of SYSPROF  EXEC     S loaded as SYSPROF  EXEC
    EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
     8:47:55 12/18/84 copy of PARSE    XEDIT    S loaded as PARSE    XEDIT
    EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
    15:40:32 10/22/85 copy of DISCARD  EXEC     S loaded as DISCARD  EXEC
    EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
    11:03:53 10/24/85 copy of NOTE     EXEC     S loaded as NOTE     EXEC
    EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
    15:41:55 10/22/85 copy of PROFNOTE XEDIT    S loaded as PROFNOTE XEDIT
    EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
    15:41:05 10/22/85 copy of ALL      XEDIT    S loaded as ALL      XEDIT
    EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
    *** End of Source List ***
    CMSINST built at 15:56:34 on 12/02/85
```

Note: When you built your CMS nucleus, if you indicated in the DMSNGP file (USEINST=YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs CMS. If you previously indicated that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild CMS. (To find the latest version of the DMSNGP file, issue SETUP to access all the minidisks where it might be; then issue FILELIST DMSNGP ASSEMBLE *. All the copies of DMSNGP ASSEMBLE will be listed, with the date when each one was last modified.)

Messages from DCSSGEN

While DCSSGEN is processing, you may receive error or warning messages that indicate specific conditions. If errors were encountered, after processing is complete you receive the following prompt:

```
DMSEXG298R An error has been detected while building the DCSS.
           Do you still want the DCSS saved?
           Enter 1 (YES) or 0 (NO).
```

Enter **1** to disregard the errors and save the segment, or enter **0** to not save the segment. If you do not save the segment, you receive the message:

DMSEXG288I *segname* not saved

If DCSSGEN encounters an error while saving the segment, you receive the message:

DMSEXG288E *dcssname* not saved

If your virtual machine is not large enough to contain the segment (you need 16M), you receive the message:

DMSEXG284E The DCSS is not completely inside the virtual machine

To correct this situation, increase the size of your virtual machine, re-IPL CMS, and reissue the DCSSGEN command.

11. Define your virtual storage **less** than the address at which the HELP segment is to be loaded. For example, if the HELP segment is defined at X'C50000' (the default address in the SAMPNSS EXEC) define your storage as 12M.

```
define storage 12m ■
Storage cleared - system reset
Storage = 12m
```

12. Re-IPL CMS:

```
ipl cms ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

13. Issue the following commands to initialize and save the segment:

```
savefd init vaddr label help ■
Ready; T=n.nn/n.nn hh:mm:ss
savefd save vaddr label help ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is the unique address of the HELP file disk for your new system default national language. (The recommended address is 19B.)

label is the CMS label assigned to the disk, for example, MNT19B.

14. For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*.

15. Verify that the CMSINST and CMSHELP segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss INSTHELP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss CMSINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss HELP        NSS      MAINT
```

16. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 6. Build a New CP Nucleus Containing the CP Language Files

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Invoke the ITASK EXEC to build the CP nucleus:

```
itask build cp noassem ■
```

If you have not already assembled HCPSYS ASSEMBLE, HCPRIO ASSEMBLE, and HCPBOX ASSEMBLE, omit the **noassem** operand.

If you assemble the files here, copy the text decks from MAINT 191 to MAINT 295.

```
nnnnnnnn FILES CHANGED
DMSACC724I 191 replaces A(191)
DMSACC724I 391 replaces D(192)
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

RDR FILE fileno TO MAINT COPY 001 NOHOLD
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The CP nucleus is being created. It will be sent to your reader.

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CP, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Order your reader so that the CP nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CP nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. Find out how much virtual storage MAINT has:

```
query virtual storage ■
STORAGE = 0016M
```

9. Perform this substep **only** if you are generating a CP nucleus with a virtual=real area:

```
define storage nnnnm ■
```

If you are generating a CP nucleus with a preferred virtual machine, MAINT's virtual storage must be at least 3 megabytes greater than that specified by the VRSIZE operand in the SYSSTORE macro instruction in HCPSYS ASSEMBLE.

10. Perform this substep **only** if you redefined MAINT's virtual storage to a size greater than 16M:

```
set machine xa ■
STORAGE CLEARED - SYSTEM RESET
SYSTEM = XA
```

Storage addresses greater than 16M require 370-XA architecture. If your machine is already in 370-XA mode, you will not get a response.

11. Load (IPL) MAINT's virtual reader:

```
ipl 00c ■

PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
MSG FROM MAINT   :
  HCPGEN350W NUCLEUS LOADED ON XASRES
  CP ENTERED; DISABLED WAIT
  PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus was successfully loaded on the system residence device. If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

If you entered a disabled wait state with a code 691, you did not observe the warning notes above.

12. Issue the following commands to ensure that these settings are in effect:

```
set machine 370 ■
STORAGE CLEARED - SYSTEM RESET
SYSTEM = 370
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 16M
```

If your machine is already in 370 mode, you will not get a response.

Saving and Printing the CP Load Map

13. The CP load map has been spooled to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
spool prt * nohold ■
close prt ■
```

The CP load map is spooled from your virtual printer to your virtual reader.

```
ipl 190 clear ■
```

After VM READ appears in the corner of your screen, issue:

```
■
access 295 d ■
query rdr * all ■
```

```
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME TYPE DIST    USERFORM OPERFORM KEEP MSG
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss          SYSPROG STANDARD STANDARD OFF  OFF
```

The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

```
receive fileno fn ft d ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1 – 9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following commands:

```
spool 00e class a ■
print fn ft d ■
```

End of Saving and Printing the CP Load Map

Step 7. Shut Down and Do a Warm Start

Note: If you have been using two terminals for the installation, go back to the console (the terminal where the OPERATOR user ID is logged on) and do this shutdown from there. After the shutdown and warm start are complete, OPERATOR is automatically logged on to the console again, and you can log on to the second terminal as MAINT.

1. Shut down the system:

```
drain all ■
```

This command stops spooling operations on all real unit record devices. Be sure to wait until all devices are drained.

```
RDR 000C DRAINED SYSTEM
PUN 000D DRAINED SYSTEM CLASS A
PUN 000D FORM STANDARD MANUAL SEP
PRT 000E DRAINING SYSTEM CLASS A
PRT 000E FORM STANDARD MANUAL SEP NO3800
PRT 000E PRINTING MAINT FILE fileno
  RECS nnnn COPY 001 SEQ 001
shutdown ■
SYSTEM SHUTDOWN STARTED
HCPGIR450W CP entered; disabled wait PSW  '000A0000 00000961'
```

First Level Only

2. IPL CLEAR the real address of your XASRES volume according to the directions in the *Operator's Guide* for your machine.

End of First Level Only

Second Level Only

3. IPL your XASRES volume:

```
ipl vdev clear ■
```

vdev is the **first-level virtual** address of your XASRES volume, which functions as a "real" address at second level.

End of Second Level Only

```
VM/XA SYSTEM PRODUCT RELEASE 21 SERVICE LEVEL 0000;
SYSTEM NUCLEUS CREATED ON mm/dd/yy AT hh:mm:ss,
LOADED FROM XASRES
******************************************
* LICENSED MATERIALS - PROPERTY OF IBM*    *
*                                          *
* 5664-308 (C) COPYRIGHT IBM CORP. 1983,   *
* 1989.  ALL RIGHTS RESERVED.              *
* US GOVERNMENT USERS RESTRICTED RIGHTS -  *
* USE, DUPLICATION OR DISCLOSURE           *
* RESTRICTED BY GSA ADP SCHEDULE CONTRACT  *
* WITH IBM CORP.                           *
*                                          *
* * TRADEMARK OF INTERNATIONAL BUSINESS    *
*   MACHINES.                              *
******************************************
⋮
HCPISU951I CP VOLID volid NOT MOUNTED
⋮
Start ((WARM|FORCE|COLD)(DRAIN)(DISABLE)(NODIRECT))|(SHUTDOWN):
warm

It is now hh:mm:ss EDT day mm/dd/yy
Change TOD clock (YES|NO):

no ■
```

If the clock is not set, consult *VM/XA SP Real System Operation* for instructions on how to set it.

```
The directory on volume XASRES at address nnnn
   has been brought online.
There is no logmsg data
FILES: NO RDR, NO PRT, NO PUN
LOGON AT hh:mm:ss EST day mm/dd/yy

HCPCPJ951I CP volid volid not mounted
⋮
```

volid is a DASD volume listed on the SYSCPVOL macro instruction of HCPSYS. If the volume is not one that you are using, ignore the message.

You are now logged onto the OPERATOR user ID.

```
STORAGE = 0016M
FILES 0000001 RDR, 0000001 PRT, NO PUN
```

4. Define the storage that you want for the OPERATOR user ID:

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
```

This is an example. You may define more than 16M.

5. Enable the devices in your revised HCPRIO file:

```
enable all ■
hh:mm:ss Command complete
terminal mode vm ■
```

6. IPL 190:

```
ipl 190 clear ■
DMSWSP327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
Ready; T=n.nn/n.nn hh:mm:ss
```

7. Log on to the MAINT user ID by doing **one** of the following:
   - Issue the DISCONNECT command to disconnect the OPERATOR user ID, then log on to the MAINT user ID at the same device,

     OR

   - Log on as MAINT at your previously defined second terminal.
8. If you have GCS installed in your system, complete Steps 8 and 9.

Step 8. Create a New GCS Configuration File (Optional)

You may want to create a new GCS configuration file for this language. For example, you may want to change the authorized user IDs or add new user IDs, or you may want to change the text of your system ID to appear in the new language.

If you do not want to create a new configuration file, go to "Step 9. Build and Save a New GCS Nucleus Containing the GCS Language Files" on page 342.

1. Access the required minidisks:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
Ready; T=n.nn/n.nn hh:mm:ss
erase 56643089 $setup a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Set up the GROUP EXEC messages in storage:

   Note: If you do not have a full-screen display device, you **cannot** use the GROUP EXEC to build the configuration file. **Do not** set up the GROUP EXEC messages in storage. Refer to the *VM/XA SP Group Control System Command and Macro Reference* and use the build macros described there to build the configuration file manually. Then continue the language installation by modifying the GCS loadlist (see page 341).

`filelist csimes[y] * * ■`	Find the latest version of CSIMES[y] TEXT or TXT*nnnnn*. The country code identifies the language of your **current** CMS system, **not** the new language that you are installing for GCS. (Of course, if you have just rebuilt CMS with a new language, they are probably the same.)
`copyfile csimes[y] [text|txtnnnnn] fm csiume[y] text a ■`	This command creates a temporary GCS message file (containing the messages for GROUP EXEC) that has the filename required by the SET LANGUAGE command.
`set language langid (add csi user ■`	*langid* identifies the language of your **current** CMS system, **not** the new language that you are installing for GCS. (Of course, if you have just rebuilt CMS with a new language, they are probably the same.) This command sets *langid* as the language of the temporary GCS message file and places the file in user storage.

3. Invoke the GROUP EXEC:

`group` *systemname* ■	This command assigns *systemname* as the filename of the GCS configuration file that you are creating and invokes the Primary Option Menu. The *systemname* parameter is optional at this time. If specified, it must match the system name in the DEFSYS entry for this GCS system in the SAMPNSS EXEC (the sample system name is GCS). If you specify a system name here, the Primary Option Menu appears with the system name filled in. If you do not specify a system name here, you must specify one on the Primary Option Menu.

Note: The GROUP EXEC panels are not shown here. For guidance, refer to the procedure that you used to install your base system. If you used the first-level Starter System procedure, see page 80. If you used the second-level Starter System procedure, see page 181. If you used the existing system procedure, see page 278.

4. After you complete the panels and exit from GROUP EXEC, remove the temporary GCS message file from user storage and erase it from your A-disk:

`set language` *langid* `(delete csi user` ■	*langid* identifies your **current** system national language, not the one your are installing.
`erase csiume[`*y*`] text a` ■	

5. Modify the GCS loadlist:

`filelist gcsload exec *` ■	Find the latest version of GCSLOAD EXEC.
`copyfile gcsload exec` *fm* `= = a (replace` ■	This command writes a copy of the loadlist to MAINT 191 and replaces the loadlist used to generate the previous GCS nucleus. **Do not** modify the source file on MAINT 595.
`xedit gcsload exec a` ■ ⋮ `file` ■	Locate the configuration file entry (&1 &2 &3 GCS) in the loadlist and change the filename from the default (GCS) to *systemname*.

Step 9. Build and Save a New GCS Nucleus Containing the GCS Language Files

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Invoke the ITASK EXEC with the BUILD GCS parameters:

```
itask build gcs systemname ■
```

systemname must match the name you specified when you issued the GROUP EXEC. If you do not specify a name, the default is GCS.

```
HCPNSD440I The Named Saved System (NSS) systemname
            was successfully defined
            in fileid fileno
OWNERID  FILETYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT

DMSUPD181E No update files were found
DMSWHM1907I Assembling systemname
DMSWHM1909I systemname TEXT A created
```

Note: If you receive an assembly error, you may have to reestablish the minidisk access order.

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■ erase 56643089 $setup a ■
```

Then go back and check your configuration file, which ITASK has renamed *systemname* ASSEMBLE. Refer to "Planning for the Group Control System (GCS)" on page 9 for information regarding required fields. Correct the file or go through the GROUP EXEC panels again to recreate the file. (If you recreate the configuration file, you must use the same *systemname*.) After you recreate the configuration file, rename or erase the old *systemname* ASSEMBLE file, then rename the new configuration file from GROUP to ASSEMBLE. Issue the ITASK BUILD GCS *systemname* command to assemble the configuration file and to build and save the nucleus.

```
PRT FILE fileno SENT FROM MAINT PRT AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
nnnnnnn FILES CHANGED
⋮
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
```

These are informational messages. You can ignore them.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 A NOHOLD NOKEEP
```

The GCS nucleus has been sent to MAINT's virtual reader.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside GCS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Copy the text deck, configuration file, ASSEMBLE file, and EXEC from MAINT 191 to MAINT 495:

```
access 495 d ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname text a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname text a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname group a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname group a ■
Ready; T=n.nn/n.nn hh:mm:ss
copy systemname assemble a = = d (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
erase systemname assemble a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Verify that the GCS nucleus is in MAINT's virtual reader:

```
query rdr maint all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

This is the file that you will IPL in substep 22 on page 88 below. It has approximately 7,600 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

6. Order your reader so that the GCS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the GCS nucleus.

7. Query the reader. If it is not class * already, spool it as class *:

```
query virtual c ■
RDR 000C CL cl NOCONT NOHOLD EOF READY
    000C nnnn           CLOSED NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
spool c class * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■
```

```
PRT FILE fileno SENT FROM MAINT PRT WAS fileno
  RECS nnnn CPY 001 NOHOLD NOKEEP
```

This message indicates that the GCS load map file has been sent to MAINT's virtual printer. If you plan to save or print the GCS load map, record *fileno*.

```
MSG FROM MAINT  :  CSIINI134I systemname has nnnnnn
   bytes of available common free storage
HCPNSS440I Named Saved System (NSS) systemname
            was successfully saved in fileid fileno
```

9. IPL CMS:

```
ipl cms ■
DMSEXG327I The installation saved segment could not be loaded
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: If necessary, purge your reader to free spool space.

Saving and Printing the GCS Load Map

10. The GCS load map has been sent to MAINT's virtual printer. To save the load map on disk, issue the following commands:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
MAINT    fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

The GCS load map is the file with a blank filename and filetype. It has approximately 2,200 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

Note the *fileno* of this file. You will use it in your next command.

```
#cp transfer prt fileno to * rdr ■
access 495 d ■
receive fileno fn ft d (replace ■
```

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1–9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print fn ft d ■
```

End of Saving and Printing the GCS Load Map

Installing Multiple GCS Systems

11. You can install more than one GCS system, if you defined the additional systems in the system directory.

    To install each additional system:

    a. Reestablish the minidisk access order:

    ```
    access 5E5 b ■
    vmfsetup 56643089 gcs (all ■
    erase 56643089 $setup a ■
    ```

    b. Set up the GROUP EXEC messages in storage (full-screen display devices only):

    ```
    copyfile csimes[y] [text|txtnnnnn] fm csiume[y] text a ■
    set language langid (add csi user ■
    ```

    The CSIMES text file is shipped as TXT*nnnnn*. The most current version of this file can be determined by referring to the first entry in the CSIMES AUX file. In the first entry, there is a UM*nnnnn* field, where *nnnnn* is the PTF number.

    c. Create a configuration file for each nucleus.

    - If you are using a full-screen display device, issue:

      ```
      group systemname ■
      ```

      systemname is a unique filename. Then complete the GCS panels to define the system parameters.

    - If you are **not** using a full-screen display device, refer to *VM/XA SP Group Control System Command and Macro Reference*. Use the build macros to create the configuration file.

    d. Remove the temporary GCS message file from user storage and erase it on your A-disk (full-screen display devices only):

    ```
    set language langid (delete csi user ■
    erase csiume[y] text a ■
    ```

    e. For each new GCS system, add a DEFSYS statement to the SAMPNSS EXEC. For information on the DEFSYS command, see the *VM/XA SP CP Command Reference*.

f. For each new GCS system, add an override section at the end of the product parameter file. This example shows an override section for a GCS system called **NEWGCS**:

```
:NEWGCS. GCS
:BLD. REPLACE
 NEWGCS VMFBDNUC BUILD1
:END.
```

You must also add the name of the new GCS system to the :OVERLST. tag at the top of the product parameter file, for example:

```
:OVERLST. CORCP CORCMS CORGCS NEWGCS
```

g. For each GCS system, assemble the configuration file, build and save the new GCS nucleus, and copy the text deck, configuration file, ASSEMBLE file, and EXEC to the 495 disk:

```
access 5E5 b ■
vmfsetup 56643089 gcs (all ■
erase 56643089 $setup a ■
itask build gcs systemname ■
access 495 d ■
copy systemname text a = = d (replace ■
erase systemname text a ■
copy systemname group a = = d (replace ■
erase systemname group a ■
copy systemname assemble a = = d (replace ■
erase systemname assemble a ■
ipl 00c clear ■
ipl cms ■
```

Before you build another GCS nucleus, save or print the GCS load map, as indicated above. Remember to give each load map file a different name.

End of Installing Multiple GCS Systems

12. Verify that all your GCS segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss GCS         NSS      MAINT
*NSS    fileno NSS  R  nnnn mm/dd hh:mm:ss systemname  NSS      MAINT
⋮
```

Part 2. Servicing the System

Part 2 of this book contains the following chapters:

- Chapter 6, "VM/XA System Product Service—An Overview" on page 349, provides an overview of the service tasks that you may need to perform. It also contains a description of the MAINT virtual machine (the user ID that you use to perform the service), and a list of the tools and EXECs you use to apply service.
- Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361, describes control files and update files, and how VM/XA SP uses them.
- Chapter 8, "Files Used in PUT and COR Service" on page 371, describes the program update tape, the corrective service tape, and the files used in applying program update service or corrective service to VM/XA SP.
- Chapter 9, "Preparing For Service" on page 415, provides a procedure for establishing an alternate level of build disks to use when servicing your system.
- Chapter 10, "Receiving Service for CMS" on page 417, provides a procedure for receiving program update service or corrective service for CMS.
- Chapter 11, "Receiving Service for CP" on page 423, provides a procedure for receiving program update service or corrective service for CP.
- Chapter 12, "Applying Service to CMS" on page 427, provides a procedure for applying program update service or corrective service to CMS.
- Chapter 13, "Applying Service to CP" on page 429, provides a procedure for applying program update service or corrective service to CP.
- Chapter 14, "Rebuilding CMS after Applying Service" on page 431, provides a step-by-step procedure for rebuilding CMS after applying program update service, corrective service, or local service.
- Chapter 15, "Rebuilding CP after Applying Service" on page 487, provides a step-by-step procedure for rebuilding CP after applying program update service, corrective service, or local service.
- Chapter 16, "Service to DV" on page 507, outlines a procedure for receiving and applying program update service or corrective service to the dump viewing facility and for rebuilding the dump viewing facility.
- Chapter 17, "Service to GCS" on page 517, outlines a procedure for receiving and applying program update service or corrective service to the group control system and for rebuilding the group control system.
- Chapter 18, "Program Update Service to Licensed Programs" on page 531, outlines a procedure for receiving and applying program update service to products that have a product service EXEC rather than a product parameter file.
- Chapter 19, "Using ServiceLink to Receive Service" on page 533, describes how to use ServiceLink to receive service.
- Chapter 20, "Receiving and Applying Local Service" on page 543, provides a step-by-step procedure for receiving and applying local service to VM/XA System Product.
- Chapter 21, "Emergency Local Service Using the Patch Facility" on page 555, provides a step-by-step procedure for applying patches to the CP or CMS nucleus.
- Chapter 22, "Removing Service from VM/XA SP" on page 559, describes how to remove service from a VM/XA System Product.

What to Do Next

If this is your first time servicing a VM/XA system, read these chapters before you begin servicing your system:

- Chapter 6, "VM/XA System Product Service—An Overview" on page 349
- Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361
- Chapter 8, "Files Used in PUT and COR Service" on page 371.

Otherwise:

- For service to CMS, CP, GCS, or DV, see Chapter 9, "Preparing For Service" on page 415.
- For service to licensed programs, see Chapter 18, "Program Update Service to Licensed Programs" on page 531.
- For local service, see Chapter 20, "Receiving and Applying Local Service" on page 543.

Chapter 6. VM/XA System Product Service—An Overview

This chapter contains:

- An introduction to service
- An overview of program update (PUT) service
- An overview of corrective (COR) service
- An overview of local service
- A description of using the MAINT virtual machine for service
- A description of the tools and EXECs used in applying local service.

Introduction

The VM/XA system as installed by the customer consists of several products, such as VM/XA System Product, PROFS, and FORTRAN. VM/XA System Product is a product which consists of the components CP, CMS, GCS, and DV. Each component consists of numerous parts, such as macros, copy files, assembler modules, text decks, executable modules, and EXECs. Some parts are serviced by complete part replacement only. These parts are sometimes called **object-maintained** code. Other parts are serviced by combining an update with the base part. These parts are sometimes called **source-maintained** code.

Changes to the system are provided as new parts, replacement parts, or changes (updates) to existing parts initially shipped on the product tape. They include:

- Updates to an assembler source file (filetype is ASSEMBLE), macro (filetype is MACRO), control block (filetype is COPY), EXEC (filetype is $EXEC), or XEDIT macro (filetype is $XEDIT)
- Replacement of existing object code (filetype is usually TXT*nnnnn*), EXECs (filetype is EXC*nnnnn*), XEDIT macros (filetype is XED*nnnnn*) HELP files, TXTLIB files, or MODULE files
- Patches to a text file (filetype is usually TXT*nnnnn*) for which there is no assembler source file.

Parts serviced by replacement require the addition of a replacement part to the system. Parts serviced by update require the careful application of an update file to the existing control structure.

Each problem that requires a fix from IBM is assigned an authorized problem analysis report (APAR) number. (The term "APAR" is ordinarily used to refer to both the fix and the report. In this book, "APAR" means the fix.) An APAR may contain service to more than one part. It may contain update service, replacement service, or both.

A program temporary fix (PTF) is a package of APARs. For update service, the PTF contains only one APAR, because each update file is a change, or "delta", to the base part. Each problem is fixed by the application of a separate delta. For replacement service, the PTF may contain more than one APAR, because replacement parts must include all previous APAR fixes in addition to the APAR fix for the problem being solved by the replacement. Figure 6 on page 350 shows three PTFs being applied to the system. PTF 1 and PTF 3 each contain a single APAR consisting of an update to Part A. PTF 2 is more complex. It contains three APARS: APAR V, APAR W, and APAR X. All three APARs require adding a replacement part to Part C. APAR X also includes updates to Part A and Part B. (PTF 2 is cumulative. Two earlier PTFs are also available, one containing only APAR V and one containing APAR V and APAR W.)

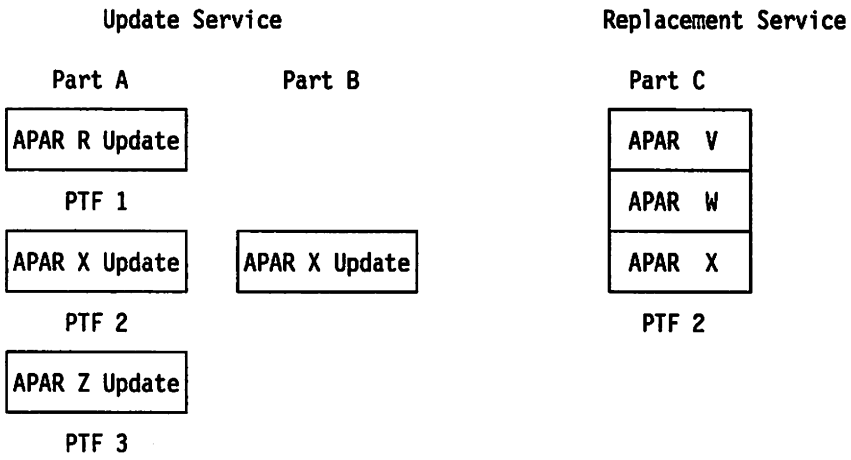


Figure 6. Update Service and Replacement Service

Service is the process of applying fixes (PTFs) to the system after installation. Problems fall into these categories:

- A PTF is available on the latest program update tape (PUT).
- A PTF is available, but no PUT is available.
- No PTF is available.

These circumstances determine the format in which the service is supplied. This, in turn, decides the procedure for applying it. These procedures are:

- Program update service:
  - A PTF is available.
  - The PTF is supplied on a PUT tape.
  - The PTF is automatically applied.
- Corrective service:
  - A PTF is available, but no PUT is available.
  - The PTF is supplied on a COR tape.
  - The PTF is automatically applied.
- Local service:
  - No PTF is available.
  - The service is originated by the customer, or is supplied by IBM on a tape other than a PUT a COR tape, in hard copy, or over the telephone.
  - The service is manually applied.
  - The service may correct the problem or circumvent it (branch around or otherwise avoid the code in which the problem occurs).

Note: These classifications have nothing to do with the classification of service as update service or replacement service. PUT service, COR service, and local service can all require either updates or replacement.

Application of PTFs on the PUT uses the procedure called program update service because its input comes from the program update tape. Massive application of PTFs on the PUT is intended to avoid problems. For this reason, program update service is sometimes called preventive service.

When a problem occurs for which a PTF does not exist on the latest PUT, IBM will provide a corrective fix if the problem is known and the PTF is ready for the next PUT. Corrective service comes on a COR tape

with the same format as the PUT tape. It is applied in essentially the same way as PUT service. A PTF can be ordered correctively even if it is already on a PUT tape.

If the problem is not known, IBM may be able to provide a temporary circumvention over the phone for a severe problem. (Circumventive service may also be provided on tape.) The circumvention usually does not fix the problem, but prevents failure by disabling the failing function. In this book, service that does not come from a PUT or COR tape is called local service, even when you get it from IBM.

Local service for replacement parts is like the procedure used for update parts, except that you do not need to assemble replacements for object code. Once updates are applied and assemblies are completed, update parts and replacement parts are the same.

For parts on replacement service there are two circumventive procedures. One procedure, the patch facility, is for the CP nucleus and CMS nucleus. The other procedure uses ZAP for CP utilities and command modules not in the nucleus.

Program Update Service Overview

All IBM customers have a Customer Profile at IBM Software Manufacturing and Distribution, or ISMD (formerly known as ISD or PID). This profile lists all the licensed program products for each customer. ISMD distributes PUT tapes that are tailored to the Customer Profile—this is how you receive PUT service for your licensed program products.

PUT service is sent out to VM/XA SP customers either on a regular basis or when you request it. It arrives in the form of a Program Update Tape (or PUT), which you apply automatically using a set of EXECs. The PUT, which may be one or more tape volumes, contains service files for one or more program products.

The PUT is cumulative. All update files up to the current level are included. For parts serviced by updates (source-maintained code), the latest text deck, which includes all APARs up to the current level, is also shipped. For parts serviced by replacement (object-maintained code), a deck is shipped for each PTF and will be reshipped for several PUT cycles.

The VMFREC EXEC is used to load PUT service for each VM/XA System Product component to the appropriate DELTA disk as specified in the product parameter file (PPF). The VMFAPPLY EXEC is used to apply it and to build the AUX control structure on the specified APPLY disk. The VMFBLD EXEC is used to build your nucleus after you receive and apply service. These EXECs are supplied on the VM/XA System Product product tape.

For products other than VM/XA System Product, the service files include a CMS service EXEC for each program product on the PUT. The service EXEC is especially designed for that program product. VMFREC maps the PUT and calls the appropriate service EXEC for each product, other than VM/XA System Product, being serviced.

The PUT also contains an EXEC called VMSERV, for compatibility. For products other than VM/XA System Product, you can use VMSERV as an alternative to VMFREC. Mount the tape on a tape drive, load the first tape file under CMS, and invoke the VMSERV EXEC. VMSERV supervises the installation of PUT service for the program products you have.

The format of each volume of the PUT is as follows:

- **The first tape file** contains the PUT DOCUMENT and the VMSERV EXEC.
- **The second tape file** contains a *Memo to Users* for each program product service on the volume.
- **The remaining tape files** contain service files for the program products on that volume.

When you invoke the VMFREC EXEC with the INFO (MEMOS option, you can load the PUT document and all memos to users to disk and print them. You must read the instructions on the PUT document and the *Memo to Users* before you continue PUT service installation.

Specific instructions for receiving and applying PUT service and for rebuilding components afterwards are found in Chapter 9 through Chapter 18.

Corrective Service Overview

| Corrective service is service IBM sends you to correct a specific problem that you or another customer has
| encountered and reported. It is sent on a corrective service tape, which has the same format as the PUT
| tape. It is applied the same way as PUT service (see "Program Update Service Overview" on page 351).

Specific instructions for receiving and applying corrective service and for rebuilding components afterwards are found in Chapter 9 through Chapter 17.

Local Service Overview

Local service is any service that is not supplied on a PUT or COR tape. It can be service that you originate, or service that is sent to you by IBM to correct or circumvent a specific problem that you have encountered and reported.

You will manually perform the following general steps for local maintenance:

1. Receive the following files to the alternate LOCAL1 disk:
   - Source update files
   - Replacement files
2. Apply the fixes by creating auxiliary control files on the alternate LOCAL1 disk
3. Build these changes into an alternate running system on the new BUILD1 disk.

| Specific instructions for receiving and applying local service are found in Chapter 20, "Receiving and
| Applying Local Service" on page 543.

| Specific instructions for rebuilding components afterwards are found in Chapter 14 through Chapter 17.

The MAINT Virtual Machine

By convention, the MAINT virtual machine is used to perform installation and service tasks. (Your installation may use a different virtual machine or machines.) MAINT's directory entry is part of the sample file, USER DIRECT, that is supplied with the starter system. The sample directory is printed in Appendix C, "VM/XA System Product Starter System Information" on page 725. You may wish to alter MAINT's virtual machine configuration, but remember that IBM created the configuration with the service process in mind.

Minidisks form an important part of MAINT's configuration. The strategy IBM follows is to separate key files onto different minidisk strings. A **string** is a set of minidisks defined in the product parameter file (see "The Product Parameter File" on page 384) for a particular use. The different strings defined in the product parameter file are:

String	Use
TASK	Service tools build disk.
LOCAL	Updates, replacement parts, and auxiliary control files for local service.
APPLY	Auxiliary control files generated by the VMFAPPLY EXEC.
DELTA	Updates and replacement parts from the PUT or COR tape.
BASE1	Object code.
BASE2	Source code.
BUILD1	The system built from the base code on the BASE disks plus the updates on the DELTA1 and LOCAL1 disks.
BUILD2-*n*	System disk extensions—exact use varies by component. For example, the HELP files are here.
SYSTEM	Minidisks belonging to other components that must be accessed.

When there is more than one minidisk in a string, the minidisks act as layers of service. The leftmost disk, as defined in the PPF, is the alternate minidisk and the rightmost disk is the production disk. Service levels are promoted (or merged) towards the production disk as service levels are tested and accepted.

MAINT's service minidisks are described in Table 10 on page 354.

Table 10 (Page 1 of 3). MAINT's Minidisks for System Service			
Comp.	**String**	**Disk**	**Contents**
CP	TASK	05E5	Service tools build disk.
	LOCAL1	0295	Update files for local service to CP.
	APPLY	0492	New CP control disk containing CP AUX files which have been generated from text files by VMFAPPLY.
		0291	Intermediate merge of CP APPLY disks.
		0292	Cumulative merge of previous levels of CP APPLY disks.
CP	DELTA1	0594	CP text files or update files containing program temporary fixes (PTFs—also known as CP service files) from the PUT.
		0591	Intermediate merge of CP DELTA1 disks.
		0294	Cumulative merge of previous levels of CP DELTA1 disks.
	BASE1	0194	CP text files, load lists, control files, and MACLIBs.
	BASE2	0394	CP source files, macro definitions, and copy files.
	BUILD1	0423	New CP minidisk of the system residence pack, used by MAINT to perform CP nucleus generation, directory maintenance, and DASD reallocation.
		0123	Current IPLable CP nucleus and system disk. Files are **not** merged from 423 to 123.
	BUILD2	049D	New mixed-case American English HELP files for CP, CMS, and GCS.
		019D	Current mixed-case American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49D to 19D.
	BUILD3	049C	New uppercase American English HELP files for CP, CMS, and GCS.
		019C	Current uppercase American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49C to 19C.
	BUILD4	0490	New CMS system disk containing CMS modules, control file, and CMS MACLIBs.
		0190	Current CMS system disk containing CMS modules, control file, and CMS MACLIBs. Files are **not** merged from 490 to 190.
	SYSTEM	0191	MAINT's work minidisk.

Table 10 (Page 2 of 3). MAINT's Minidisks for System Service

Comp.	String	Disk	Contents
CMS, DV	TASK	05E5	Service tools build disk.
	LOCAL1	0395	Update files for local service to CMS.
	APPLY	0692	New CMS control disk containing CMS AUX files which have been generated from text files by VMFAPPLY.
		0391	Intermediate merge of CMS APPLY disks.
		0392	Cumulative merge of previous levels of CMS APPLY disks.
	DELTA1	0593	CMS text files or update files containing program temporary fixes (PTFs—also known as CMS service files) from the PUT.
		0691	Intermediate merge of CMS DELTA1 disks.
		0293	Cumulative merge of previous levels of CMS DELTA1 disks.
	BASE1	0193	CMS text files, CMS generation and utility tools, IOCP, and the dump viewing facility.
	BASE2	0393	CMS source files, macro definitions, and control blocks.
CMS, DV	BUILD1	0490	New CMS system disk containing CMS modules, control file, and CMS MACLIBs.
		0190	Current CMS system disk containing CMS modules, control file, and CMS MACLIBs. Files are **not** merged from 490 to 190.
	BUILD2	049D	New mixed-case American English HELP files for CP, CMS, and GCS.
		019D	Current mixed-case American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49D to 19D.
	BUILD3	0501	New CMS system disk for EREP files.
	BUILD4	049C	New uppercase American English HELP files for CP, CMS, and GCS.
		019C	Current uppercase American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49C to 19C.
	SYSTEM	0191	MAINT's work minidisk.
		0295	CP LOCAL1 minidisk.
		0594	CP alternate DELTA1 minidisk.
		0591	CP intermediate alternate DELTA1 minidisk.
		0294	CP current DELTA1 minidisk.
		0194	CP BASE1 minidisk.

Note: For service to the CMS and DV components, the CP minidisks on the SYSTEM string are required in order to access the CP macro libraries.

Table 10 (Page 3 of 3). MAINT's Minidisks for System Service			
Comp.	**String**	**Disk**	**Contents**
GCS	TASK	05E5	Service tools build disk.
	LOCAL1	0495	Update files for local service to GCS.
	APPLY	0892	New GCS AUX files which have been generated by VMFAPPLY and UPDATE shells created by VMFREC.
		0491	Intermediate merge of GCS APPLY disks.
		0592	GCS current cumulative merge of AUX files which have been generated by VMFAPPLY and UPDATE shells created by VMFREC.
GCS	DELTA1	0896	New GCS text files, update files and AUX files.
		0791	Intermediate merge of GCS DELTA1 disks.
		0596	GCS current cumulative merge of text files and update files from previous levels of PUT.
	BASE1	0595	GCS text files, load lists, control files, and MACLIBs.
	BUILD1	0895	New GCS system.
		0595	Current GCS system.
	BUILD2	089E	New GCS system extensions.
		059E	Current GCS system extensions.
	BUILD3	049D	New mixed-case American English HELP files for CP, CMS, and GCS.
		019D	Current mixed-case American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49D to 19D.
	BUILD4	049C	New uppercase American English HELP files for CP, CMS, and GCS.
		019C	Current uppercase American English HELP files for CP, CMS, and GCS. Files are **not** merged from 49C to 19C.
	BUILD5	0490	New CMS system disk containing CMS modules, control file, and CMS MACLIBs.
		0190	Current CMS system disk containing CMS modules, control file, and CMS MACLIBs. Files are **not** merged from 490 to 190.
	SYSTEM	0191	MAINT's work minidisk.

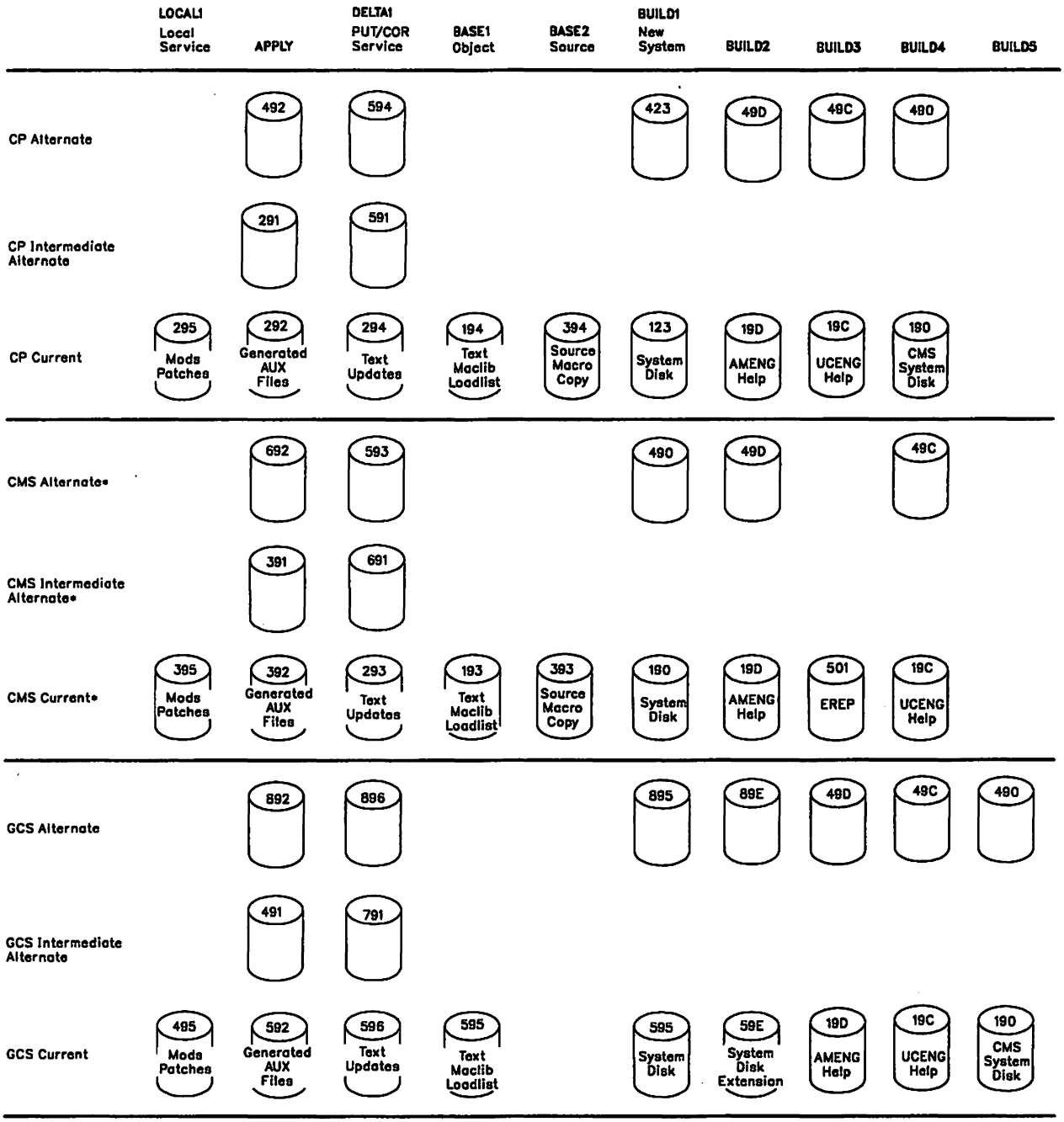


Figure 7. Alternate, Intermediate Alternate, and Current Minidisks for System Service

MAINT's minidisk configuration allows a multilevel updating scheme that preserves the data integrity of the source code and allows you to build a system with the latest updates. The organization of MAINT's minidisks is shown in Figure 7. This layout allows you to build a new system on the alternate minidisks without altering your running system and lets you back out to the old system if necessary. When you are satisfied with your new system, the alternate disks are copied to the current system's disks so that the alternate disks can be used to build your next system.

The disk layout in Figure 7 on page 357 is created on the following principles:

1. Never change anything IBM sends you. To be sure this does not happen, never write-link to a disk you do not want to write on. The BASE disks can be read-only during service. The DELTA disks can be read-only except when VMFREC and VMFAPPLY are running.
2. Keep components separate from each other, so that (for example) changes to CP will not affect CMS.
3. Keep locally applied service separate from IBM service, so that changes to the LOCAL disk will not inadvertently change IBM service on the DELTA disk.
4. Keep base release installation separate from service, so that applying service will not inadvertently change the original product as installed on the BASE disks.
5. Keep alternates for DELTA disks so that you can back out.

   Plan for backout in case something goes wrong when you apply service. Alternates make reconstruction of the previous level easier.
6. Keep an alternate BUILD disk.

   Do not change the system while it is running. An alternate BUILD disk minimizes the amount of reconstruction necessary to back out.

Tools and EXECs Used During the Service Process

You need to use many different programs and EXECs during the service procedure. Below is a partial list of the programs and EXECs you will use:

VMFREC Use VMFREC to load product service from a tape to your system. This EXEC is required for VM/XA System Product service. It is compatible with VMSERV and existing product service EXECs. You can use it to load the tape document and all Memos to Users for that tape as well as to map the tape. The map is compatible with VMSERV.

For more information on VMFREC, see "VMFREC EXEC" on page 691.

VMFAPPLY Use VMFAPPLY with the APPLY list to create AUX files for those PTFs being added to your system.

For more information on VMFAPPLY, see "VMFAPPLY EXEC" on page 645.

VMFMAC The VMFMAC EXEC updates and rebuilds macro libraries.

For more information on VMFMAC, see "VMFMAC EXEC" on page 665.

VMFNLS The VMFNLS EXEC updates national language-related files (for example, message repositories).

For more information on VMFNLS, see "VMFNLS EXEC" on page 672.

VMFHASM VMFHASM invokes the UPDATE command to automatically incorporate update files for a particular ASSEMBLE file into a temporary copy of the ASSEMBLE file, then invokes the H Assembler to assemble the copy. The assembled files (called TEXT files) are used when the system is regenerated by the VMFBLD EXEC.

For more information on VMFHASM, see "VMFHASM EXEC" on page 657.

VMFBLD Use VMFBLD to build the nucleus parts of VM/XA System Product.

For more information on VMFBLD, see "VMFBLD EXEC" on page 652.

VMFOVER The VMFOVER EXEC makes a temporary copy of one component section from the product parameter file with changes made in accordance with the override section of the product parameter file or a separate PPF override file. The other service and installation EXECs use this copy instead of the base product parameter file.

For more information on VMFOVER, see "VMFOVER EXEC" on page 677.

VMFPLC VMFPLC calls either VMFPLCD or VMFPLC2. VMFPLC supports the VMFPLCD and VMFPLC2 command formats. A ROUTE function allows VMFPLC to establish command routing for subsequent VMFPLCD or VMFPLC2 commands issued by VMFPLC.

For more information on VMFPLC, see "VMFPLC EXEC" on page 679.

VMFPLCD VMFPLCD loads files from an envelope file, dumps CMS files to the envelope file, loads previously dumped files to the envelope file, and performs control operations on the envelope file.

For more information on VMFPLCD, see "VMFPLCD EXEC" on page 681.

VMFPLC2 VMFPLC2 loads all or a specified number of files from tape. Directions for its use as a command appear in the PUT document; it is also automatically invoked by VMFREC, VMSERV, and product service EXECs.

For more information on VMFPLC2, see "VMFPLC2 Command" on page 687.

VMFSETUP The VMFSETUP EXEC is invoked by the other service EXECs to set up the minidisk access order.

For more information on VMFSETUP, see "VMFSETUP EXEC" on page 700.

VMFVIEW The VMFVIEW EXEC invokes XEDIT to allow you to view the exception logs. Using VMFVIEW's PF key assignments, you can view all the messages of a specific type, all the messages of a specific number, the HELP screen for a particular message, and move backwards and forwards through the displayed exception log.

For more information on VMFVIEW, see "VMFVIEW EXEC" on page 702.

VMSERV VMFREC has superseded VMSERV, but you may continue to use the VMSERV EXEC to install products other than VM/XA System Product from the PUT. This EXEC is in the first file on a PUT; it can be used to print the PUT document and all memos to users for that PUT.

XEDIT XEDIT is the CMS editor shipped with VM/XA System Product. Use it to create control files in the local service procedure.

If you need information on using XEDIT, see *VM/XA SP System Product Editor Command and Macro Reference*, or *VM/XA SP System Product Editor User's Guide*.

Chapter 7. How VM/XA System Product Uses Control Files and Update Files

This chapter describes:

- What control files, auxiliary control files, and update files are
- File-naming conventions used in VM/XA SP service procedures
- How different control files can be used to create different systems.

The VM/XA System Product service process relies heavily on CMS files called control files. Each component of VM/XA System Product has at least one control file structure.

There are control files, auxiliary control files, and the actual update files (also called service files). These files may have the following generic filetypes:

Filetype	File Contents
CNTRL	Control file.
AUX*xxxxx*	Auxiliary control file. *xxxxx* is up to 5 alphabetic or numeric characters.
UPDT*xxxx*	Update (listed in a CNTRL file; this is not recommended because it makes tracking difficult). *xxxx* is up to four alphabetic or numeric characters.
A*mmmmmxx*	Update (listed in an AUX file). *mmmmm* is an APAR number and *xx* is HP (for CP and dump viewing facility assembler source and object updates), PP (for other CP and dump viewing facility object updates), DS (for CMS modules), or CI (for GCS).
E*mmmmm*DS	CMS macro update (listed in an AUX file). *mmmmm* is an APAR number.

Control files may be extremely simple or quite elaborate, but their important function is to allow you to control those changes necessary to tailor your system to meet your current needs. Each structure of control files is a tree with the main control file at the base, auxiliary control files branching off from the main control file, and update files branching off from the auxiliary control files.

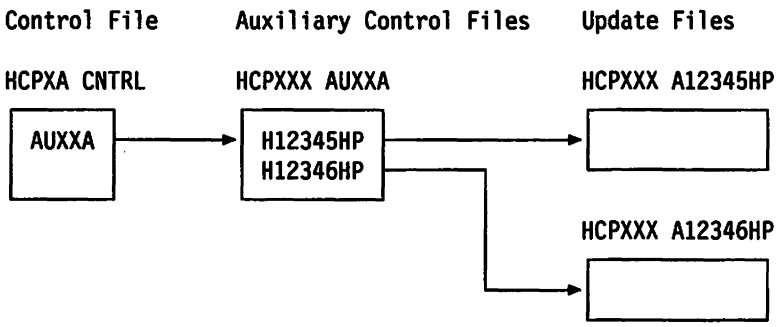


Figure 8. Service File Structure

Figure 8 shows the main CP control file, together with an auxiliary control file and its corresponding update files. (There are actually many auxiliary control files with update files). The control file HCPXA CNTRL contains several lines, one of which is the filetype of the VM/XA SP auxiliary control files (AUXXA). Every assemble file, macro, control block file, EXEC, or XEDIT macro, that has had service applied to it has a corresponding auxiliary control file called *fn* AUX*xxxxx*, where *fn* is the filename of the file that has been

serviced. (For example, the auxiliary control file for PUT service to HCPXXX ASSEMBLE is HCPXXX AUXXA.) Each auxiliary control file, in turn, contains the filetypes of the update files that have been applied.

Figure 8 on page 361 applies to all components of VM/XA SP. The structure for all components is identical, except that the main control filename and update file suffix vary (for example, the CMS main control file is called DMSXA CNTRL instead of HCPXA CNTRL; and the update file suffix is DS).

Control Files

Control files are used in:

- Updating assembler source files and EXECs using XEDIT
- Updating control blocks and macros using XEDIT
- Creating and updating AUX files using VMFAPPLY
- Assembling source code using the VMFHASM EXEC procedure
- Patching object code using the PATCH facility
- Building MACLIBs using the VMFMAC EXEC procedure
- Regenerating updated EXECs using EXECUPDT
- Creating an executable CP or CMS nucleus using VMFBLD
- Generating CP utilities using UTILITY
- Generating CMS modules using CMSGEND.

Control files are used by the CMS UPDATE command. The EXECUPDT, XEDIT, VMFMAC, and VMFHASM procedures invoke UPDATE with the CTL option to modify source files. The VMFAPPLY, VMFBLD, and HCPLDR programs also use the control file structure directly; usually the same control file is used by all these procedures for a given product or component. For the purposes of product service, all the procedures require that the control file must have a filetype of CNTRL. The control filename is unique for each VM/SP XA component. The VM product control filenames and their contents are provided in "Main Control Files" on page 364.

Sample Control File

For an understanding of how the update procedures work, you should be familiar with the elements in a control file. A sample control file for the VM/XA System Product system is shown in Figure 9.

```
TEXT MACS  HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT AUXPAT TX$   * Local Patches
LCL AUXLCL LCL   * Local Modifications
TEXT AUXXA       * IBM Put Service, COR service, and SUP reach-ahead
```

Figure 9. Sample Control File – HCPXA CNTRL

There are usually three types of records in a control file: MACS records, AUX file identification records, and comments. (A fourth type, update file identification records, will work but should be avoided.)

MACS Records

A control file must have at least one MACS record. It must appear before any AUX file identification records. The first field in the MACS record is the update level identifier. You can think of this as a name field. If no update files are found, the filetype of the assembled text deck will be derived from the update level identifier on the MACS record: TEXT if the update level identifier is TEXT and TXT*xxxxx*, where *xxxxx* is the update level identifier, if it is anything else.

| The remaining fields in the MACS record are the names of macro libraries. VMFHASM uses the library list
| from the MACS record to issue a GLOBAL command before assembling the updated source file. The
| number of macro libraries on a MACS record is limited by the number that can fit on an 80-character line.
| The total number of macro libraries specified on all the MACs records cannot exceed 63. The libraries are
| searched in the order specified on the MACS record.

AUX File Identification Records

A control file can have any number of AUX file identification (AUX*xxxxx*) records. The first field in an AUX file identification record is the update level identifier. If the auxiliary control file named in this record is found, but the top entry in the auxiliary control file contains no PTF number or local tracking number, the filetype of the assembled text deck will be derived from the update level identifier on the AUX file identification record: TEXT if the update level identifier is TEXT and TXT*xxxxx*, where *xxxxx* is the update level identifier, if it is anything else. (If more than one auxiliary control file is listed, the last one found—the one named on the top auxiliary control file identification record—determines the text deck filetype.)

The second field (sometimes called the update ID) is the filetype of the auxiliary control file. The characters AUX identify an auxiliary control file that lists additional fixes to be applied; AUXLCL in this example. AUXXA is the VM/XA System Product auxiliary control file, listing updates distributed by IBM. This file is listed at the bottom of the control file so that these updates are applied first. (Note that the file is read from the bottom to the top by UPDATE and from the top to the bottom by VMFBLD. UPDATE applies IBM changes first and local changes last, while VMFHASM names the text file, and VMFBLD finds it, according to the last change applied.)

Note: Service supplied by IBM for VM/XA System Product does not use preferred AUX files, but they are supported. If you wish to use them, see "Preferred AUX File" on page 639.

| The preferred AUX file identifier is an optional field. There can be 1 to *n* preferred AUX files. Each token
| in this field must be 4 to 8 characters in length. Preferred AUX files are only valid when the UPDT/AUX
| file identifier specifies an AUX file. If a preferred AUX file exists, the line in the control file with the
| preferred AUX file is ignored and the line above it is used. If a preferred AUX file does not exist, it uses the
| line in the control file. The file specified by the preferred AUX file identifier does not have to be an AUX
| file, it can be as update file as well.

| The text deck prefix is an optional field. It can come before or after the patch indicator field. The text deck
| prefix must be 3 characters in length. This field is used to derive the filetypes of text decks. If this field is
| not present, the default **TXT** is used. The text deck prefix cannot be **TXS** or **AUX**.

| The patch indicator is an optional field. It can come before or after the text deck prefix field. This field
| simply indicates that the AUX file specified by the UPDT/AUX file identification field may contain patch
| updates. If this field is not present on an UPDT/AUX file identification record, patch updates are not
| permitted for this record.

| If the characters **PTF** appear in the level identifier field, the UPDATE command will take the second field as
| the filetype of an update file even if it begins with **AUX**.

Comments

A control file can have any number of comment records. Comment records are identified by an asterisk (*) in the first column. They may appear anywhere in the control file. Comments may also appear at the end of control records. They begin with an asterisk.

For the UPDATE command only, any field that follows the UPDT/AUX file identifier field that is less than four or more than eight characters long is taken to be the beginning of the comment field. This is possible because UPDATE does not use the text deck prefix or the patch indicator fields.

Update File Identification Records

For compatibility with other VM systems, VM/XA System Product also supports update file identification records in control files. Avoid using them if possible, because an update listed in a control file must be called UPDT*xxxx*. The four variable characters are not enough to identify the APAR with which this update is associated, essential information for tracking. List updates in an auxiliary control file, where the name can include an APAR number.

Varying Control Files to Generate Multiple Systems

By varying the control files, you can build different versions of your system from a single set of AUX and update files. For example, you might have the following files:

- PRODSYS CNTRL:

```
*THIS IS A SAMPLE CNTRL FILE
*DO NOT USE THIS FILE
TEXT MACS HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
LC2 AUXLCL2 LCL TX$
P1 AUXP1
TEXT AUXXA
```

- TESTSYS CNTRL:

```
*THIS IS A SAMPLE CNTRL FILE
*DO NOT USE THIS FILE
TEXT MACS TESTLIB HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
LC1 AUXLCL1 LCT TX$ * FREE FORM COMMENT
LC2 AUXLCL2 LCL TX$
P1 AUXP1
TEXT AUXXA
```

Notes:

1. The control file for the test system contains one more level of updates (the record beginning with LC1) than the control file for the production system. When all the updates listed in the auxiliary control files have been applied, the system built with the test control file will contain more updated files than the system built with the production control file. Both control files serve as the base of a tree using the same set of AUX and update files, but the test tree has branches not in the production tree. It is not necessary to keep two sets of AUX and update files.
2. The test system includes one more macro library (TESTLIB) than the production system.

Main Control Files

CP, CMS, CMS macros, DV, GCS, and the VM/XA SP loader each have a main control file. The contents of these main control files are shown below.

The contents of HCPXA CNTRL (control file for CP) are:

```
TEXT MACS  HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT AUXPAT TX$   * Local Patches
LCL AUXLCL LCL   * Local Modifications
TEXT AUXXA       * IBM Put Service, COR service, and SUP reach-ahead
```

The contents of DMSXA CNTRL (control file for CMS) are:

```
TEXT MACS DMSGPI DMSOM OSMACRO CPLIB
PAT AUXPAT TX$    * Local Patches
LCL AUXLCL LCL    * Local Modifications
TEXT AUXXA        * IBM Put Service, COR service, and SUP reach-ahead
```

The contents of DMSMXA CNTRL (control file for CMS macros) are:

```
TEXT MACS
PAT AUXMPAT TX$   * Local Patches
LCL AUXMLCL LCL   * Local Modifications
TEXT AUXXA        * IBM Put Service, COR service, and SUP reach-ahead
```

The contents of HCSXA CNTRL (control file for dump viewing facility) are:

```
TEXT MACS  HCSOCO HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
PAT AUXPAT TX$    * Local Patches
LCL AUXLCL LCL    * Local Modifications
TEXT AUXXA        * IBM Put Service, COR service, and SUP reach-ahead
```

The contents of CSIXA CNTRL (control file for GCS) are:

```
TEXT MACS CSIGPI CSIOM DMSGPI DMSOM
PAT AUXPAT TX$    * LOCAL PATCHES
LCL AUXLCL LCL    * LOCAL MODIFICATIONS
TEXT AUXXA        * IBM Put Service, COR service, and SUP reach-ahead
```

The contents of HCPLDRCM CNTRL (control file for the VM/XA SP loader) are:

```
TEXT MACS HCPLDRCM HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
TEXT AUXCMS
TEXT AUXXA
```

Notes:

1. HCPLDRCM CNTRL is used only to assemble the VM/XA SP loader.
2. All the service EXECs obtain the control filename from the product parameter file, except UTILITY EXEC, which always uses HCPXA CNTRL. HCPXA CNTRL is the usual control file for CP and for utilities.

Local Control Files

The IBM-supplied control files already provide for one level of local service; but when you create files for local service or updates of VM/XA System Product modules, you may still need to create a local control file. A local control file is a copy of the appropriate VM/XA System Product control file with an entry for each local auxiliary control file and the filename of your local MACLIB. For example, the file CPLCL CNTRL may contain:

```
    TEXT MACS YOURMAC HCPGPI HCPPSI HCPOM1 HCPOM2 CPLIB DMSGPI DMSOM OSMACRO
    LCT AUXLCT LCT  * TEST UPDATES
    LCL AUXLCL LCL
    TEXT AUXXA
```

Note: Control files (CNTRL) and auxiliary control files (AUX*xxxxx*) are read from the bottom up when updates are applied. The IBM-supplied auxiliary files should be the bottom entry in the control file so that during assembly the IBM updates are applied first.

Auxiliary Control Files

When auxiliary control files are distributed by IBM for VM/XA System Product, they have the filetype AUXXA. If an auxiliary control file is not distributed, it must be created by VMFAPPLY. Figure 10 on page 366 shows a sample auxiliary control file for source updates. Figure 11 on page 366 shows a sample auxiliary control file for patches to text decks.

```
Update ft
(APAR #)     PUT Level   PTF #      Comments
A12567HP     201         UM98765    * COMMENT DESCRIBING FIX
```

Figure 10. Sample Auxiliary Control File for Source Updates

```
Update ft    Patch
(Patch #)    Indicator   APAR #     Comments
P23877       TX$         VM23877    * A PATCH TO BE REPLACED BY APAR VM23877
```

Figure 11. Sample Auxiliary Control File for Patches to Text Decks

For local source updates, a local tracking number takes the place of the PTF number in the auxiliary control file. Like the PTF number, a local tracking number has seven characters (two letters followed by five numbers).

When the text deck filetype is created, the two letters of the PTF number or local tracking number are ignored. The five numbers are appended to the text deck filetype prefix if one is present in the main control file. If no text deck filetype prefix is found, TXT is used as the prefix.

Update Files

Update files can contain:

- Updates to source code
- Patches to object code
- Requisite information.

All of the PTF and update files distributed by VM/XA System Product are assigned filetypes as follows:

Update File	Built Parts	APAR Number	PTF Number
A*mmmmm*HP	TXT*nnnnn*	VM*mmmmm*	UM*nnnnn*
A*mmmmm*PP	EXC*nnnnn*		
A*mmmmm*DS	XED*nnnnn*		
A*mmmmm*CI	⋮		
E*mmmmm*DS			

Where:

A
indicates a VM/XA SP update.

E
indicates a VM/XA SP CMS macro update.

HP
is the 2-character identifier for VM/XA System Product CP and dump viewing facility assembler source and object updates.

PP
is the 2-character identifier for other VM/XA System Product CP and dump viewing facility updates.

DS
is the 2-character identifier for VM/XA System Product CMS.

CI
is the 2-character identifier for VM/XA System Product GCS.

mmmmm
is an APAR number. You can enter the command FILELIST * **mmmmm** * to see a listing of all the modules affected by APAR *mmmmm*. The filename is the same as the module name. Browse the AUX file for any one of these modules to find the corresponding PTF number.

nnnnn
is a PTF number. You can enter the command FILELIST * **nnnnn** * to see a listing of all the modules affected by PTF *nnnnn*. The filename is the same as the module name. Browse the AUX file for any one of these modules to find the corresponding APAR number.

For example, the code and fixes to answer APAR VM12567 against the CP module HCPCFM in VM/XA System Product are contained in the file HCPCFM A12567HP.

The file HCPCFM AUXXA contains the entry:

```
A12567HP    201       UM98765  *COMMENT DESCRIBING FIX
```

UM98765 is the PTF corresponding to APAR VM12567. If UM98765 is the last PTF applied to HCPCFM, the assembled text deck will be called HCPCFM TXT98765.

For your local updates, assign a local tracking number with the format LC*nnnnn*, where the prefix LC indicates a local fix. The five numeric digits of the local tracking number can be used to name text decks in the same way as the five numeric digits of the PTF number.

Source Update Files

Source update files are named *filename XnnnnnYY*, where:

- *filename* is the name of the part being serviced
- *nnnnn* is the APAR number
- *X/YY* is the value assigned to the :SLVI. tag in the product parameter file.

Each source update file contains:

- Requisite information, consisting of PREREQ, CO-REQ, and IF-REQ entries
- DEPEND entries added by VMFAPPLY, identifying any PTFs for which this PTF is a requisite
- Update control statements and assembler language statements defining an effective change to a module's source code. (These changes are actually made as object code is created.)

Source update files are shipped on the PUT or COR tape. When a source update is applied, VMFAPPLY changes the filemode of the update file from 1 to 5. Figure 12 is an example of a source update file.

```
./ * PREREQ: VM33332 VM31452
./ * CO-REQ: NONE
./ * IF-REQ: NONE
./ * DEPEND: A33330DS A34594DS
./ I 09180000          $ 9182000 2000          08/16/88 20:56:22
         TM    MISFLAG2,OSTYPLD WAS THIS AN OS TYPE LOAD?      @VA33331
*                           (WERE WE CALLED BY DMSSLN?)       @VA33331
         BO    LDRB0231       Don't reset EPA to new value     @VA33331
./ I 09190000          $ 9190500 500           08/16/88 20:56:22
LDRB0231 DS    0H                                             @VA33331
```

Figure 12. Example of a Source Update File – DMSLDR A33331DS

Patch Update Files

The customer creates patch update files for parts of the CP nucleus, CMS nucleus, and GCS nucleus serviced by replacement; that is, parts for which no source code is available. Patch update files contain:

- Requisite information, consisting of PREREQ, CO-REQ, and optional IF-REQ entries.
- VERIFY and REPLACE statements that effectively change the object code as the nucleus is created. When you apply a patch update, change the filemode of the update file from 1 to 5. Figure 13 is an example of a patch update file.

```
./ * * PREREQ: NONE
./ * * CO-REQ: NONE
./ * NAME HCPAFF
./ * * This is a patch.  It
./ * *   will be replaced by
./ * *   APAR VM23877.
./ * VER 61C 0780
./ * REP 61C 0700 ■
```

Figure 13. A Sample Patch Update File

Update Shells

VMFREC creates update shells on the APPLY disk for parts on replacement service. Update shells use the same naming convention as update files. An update shell takes the place of the source update file that would exist if the part were on update service. Update shells contain requisite information and DEPEND entries (created by VMFAPPLY), but no executable code. Like source update files, update shells have the module name for a filename and a filetype derived from the APAR number. When the PTF associated with the APAR is applied, VMFAPPLY changes the filemode of the update shell from 1 to 5. Figure 14 is an example of an update file shell.

```
./ * PREREQ: NONE
./ * CO-REQ: NONE
./ * IF-REQ: NONE
./ * * THIS IS AN UPDATE FILE SHELL
```

Figure 14. Example of an Update File Shell – DMSMGC A34362DS

Guidelines for Using Update Files

1. Keep each fix (patch or APAR) in a separate update file. This applies to both source updates and patch fixes. Several files may be required to make a single fix (this happens when the same fix applies to several parts), but you should never have several fixes in the same file.

   Each fix should have a unique identifier for control purposes. This identifier is the filetype of the update file. If the same fix applies to several parts, there should be a file for each part, all with the same filetype.

2. Keep all local fix descriptions for the same part in the same AUX file, unless a fix applies to a different control file level.

   Local fixes for the same part should not be distributed over AUX files (different control file levels) arbitrarily. Local service should be easily distinguished from IBM service and should always be applied last. Local service can be distributed over separate control files for the purpose of maintaining different service levels with a single structure of AUX and update files. Each level can be built from a different control file containing only the desired level identifiers.

3. Never place local patches in AUX files from IBM. In other words, keep your local service separate from IBM service. Local service should be easily distinguished from IBM service and should always be applied last.

4. Patches to text files should be applied only when no source file is available. If you do apply text file patches when source code is available, they should be converted to source updates and reassembled before they are moved from a test environment to a production-level system.

   Building local source updates on top of local text file patches for the same part will lead to confusion.

5. Do not place the names of update files directly in the main control file. Place update filenames in an auxiliary control file.

   If you place the update filename in the main control file instead of an auxiliary control file, the update file's filetype must be UPDT*xxxx*. If you place it in an auxiliary control file, the update file's filetype can be derived from the APAR number, and you can specify the text deck's filetype (derived from the PTF number). These filetypes are used during assembly and nucleus build.

Chapter 8. Files Used in PUT and COR Service

This chapter describes:

- The files used in applying PUT and COR service
- The program update tape (PUT)
- The COR tape.

Files Used in PUT and COR Service

Besides the source or object code to be updated, the following files are used to apply PUT and COR service:

- PUT document
- COR document
- PUT descriptor file
- COR descriptor file
- Product contents directory
- Memo to Users
- Text decks
- Text shells
- PTF parts list
- Apply list
- Exclude list
- Exception log
- Receive history file
- Load list
- Temporary load list
- Load map
- Restart indicator files
- Product parameter file
- Product parameter override file
- Temporary product parameter file
- Service disk map.

The Tape Document

- PUT DOCUMENT

  The PUT document describes the service procedure for the PUT tape. It lists the steps involved in applying service to the system, and contains a description of the VMSERV EXEC. (Other service installation EXECs are described in the Memo to Users.) Read the PUT document before you try to apply service.

- COR DOCUMENT

  The COR document describes the service procedure for the COR tape. It lists the steps involved in applying service to the system, and contains an explanation of the service EXECs. Read the COR document before you try to apply service.

The Tape Descriptor File

- PUT Descriptor File - PUT *nnnnn*

  The PUT descriptor file is a directory of the products and components for which service is contained on the program update tape. It shows how many files are available for each product and component and where they are. The PUT descriptor file is called PUT *nnnn*, where *nnnn* is the PUT number. Figure 15 on page 373 is an example of the tape descriptor file.

  The first record of the PUT descriptor file states that it applies to a PUT tape. (This record distinguishes the PUT descriptor file from the COR descriptor file described below.) The second record tells how many volumes the tape has and which volume this is. If there is more than one volume, the PUT descriptor file appears on each volume, but describes the whole tape.

  The other records are the multivolume directory. A :VOL*nn* record indicates the beginning of each volume. Each of the 3-word records following a :VOL*nn*. record describes a group of tape files. (A tape file is the data between two tape marks. It may contain more than one CMS file.) If the first two words are PUT FILES, that group of files is the two standard tape files that begin every volume of the PUT tape: one containing the PUT descriptor file and PUT DOCUMENT, and the other containing informational files about all the products for which service is supplied on the tape (see Figure 29 on page 412). Otherwise, the first word names the product to which the files apply; the second word is either HDR (meaning a header file) or the name of the component to which the files apply. The third word is always the number of files in the group.

  Note: A product may be entirely on a single volume (*prodid1*); that it may cross volumes at a component break (*prodid2*); or that a single component may span multiple volumes (*prodid3 comp2*). In the case of a product (or component) crossing a volume boundary, the break will be indicated in the product contents directory by the inclusion of a :VOL*nn* record, where *nn* indicates the continuation volume.

- COR Descriptor File - COR *ymdd*

  The COR descriptor file is a directory of the products and components for which service is contained on the corrective service tape. It shows how many files are available for each product and component and where they are. The COR descriptor file is called COR *ymdd*, where *y* is the last digit of the year, *m* is the month (numbered 1 – C in hexadecimal), and *dd* is the day; for example, 8B05 is November 5, 1988. The COR descriptor file has the same format as the PUT descriptor file (see Figure 15 on page 373).

```
PUT nnnn                              COR ymdd

VM System Program Update Tape         VM Corrective Service Tape 9110 1X660 PSIP SP2NP31
                                        89/01/10 14:02:04.157208
VOLnn of 03                           VOL01 of 01
:VOL01.                                  :VOL01
PUT   FILES     02                             COR      FILES    02
prodid1 HDR     01                    56643089 HDR      01
prodid1 comp1   nn                    56643089 CP       08
prodid1 comp2   nn                    56643089 CMS      09

prodid1 comp3   nn                    56643089 DV       07
prodid2 HDR     01                    56643089 GCS      03
prodid2 comp1   nn
prodid2 comp2   nn
:VOL02.
PUT   FILES     02
prodid2 HDR     01
prodid2 comp3   nn
prodid2 comp4   nn
prodid3 HDR     01
prodid3 comp1   nn
prodid3 comp2   nn
:VOL03.
PUT   FILES     02
prodid3 HDR     01
prodid3 comp2   nn
prodid3 comp3   nn
```

Figure 15. Example of Tape Descriptor Files

The Product Contents Directory

There is a product contents directory for each new format product serviced on the PUT or COR tape. The product contents directory lists the tape files supplied for each component of each product, and must be repeated if the product crosses a tape boundary. The product contents directory can be seen as an expansion of the PUT descriptor file or COR descriptor file: the descriptor file tells you how many files you have and where they are; the product contents directory tells you what the files listed in the descriptor file are. Figure 16 on page 374 is an example of a product contents directory.

The filename of the product contents directory is the product ID. The filetype is:

- On the PUT tape: $PUT*nnnn*, where *nnnn* is the PUT number
- On the COR tape: $COR*ymdd*, where *y* is the last digit of the year, *m* is the month (numbered 1 – C in hexadecimal), and *dd* is the day; for example, 8B05 is November 5, 1988.

prodid1 **$PUT***nnnn*

```
:VOL01.
:CP.
AXLIST
PARTLST
UPDT
TEXT
MACAUX
MACUPDT
SOURCE
MACLIB
EXEC
RESV

:CMS.
AXLIST
PARTLST
UPDT
TEXT
MACAUX
MACRO
MACUPDT
SOURCE

:VOL02.
MACLIB
EXEC
MODULE
HELP
IOCP
RESV

:DV.
AXLIST
PARTLST
DVFTEXT
DVFMODS
RESV
```

prodid1 **$COR***ymdd*

```
:VOL01
:CP.
AXLIST
PARTLST
UPDT
TEXT
MACAUX
MACUPDT
MACRO

:CMS
AXLIST
PARTLST
UPDT
TEXT
MODULE
```

Figure 16. Examples of Product Contents Directories

The Memo to Users – 56643089 MEMO

You'll receive a *Memo to Users* for each licensed program that needs service. The Memo contains instructions for servicing each program, and describes the service installation EXEC (if any) for each program. You should read the Memo for each licensed program before you try to apply service.

The *Memo to Users* for VM/XA System Product Release 2.1 is called 56643089 MEMO.

Text Decks

The self-documenting text decks for CP and CMS service contain a prolog with the following information:

1. A description of the APAR, including the APAR number, PUT level, PTF number, and content (from the AUX file)
2. Prerequisites and corequisites for this service, consisting of PREREQ, CO-REQ, and IF-REQ entries and agreeing with the requisites in the update file and update shell
3. A date and time stamp
4. The macro libraries used (optional).

The text decks also contain assembler text consisting of ESD, TXT, RLD, and END cards.

If the filetype of an update file listed in a text deck begins with a percent sign (%), that update is listed in the control file and not in an auxiliary control file. Figure 17 is an example of a text deck.

```
A33398DS 203 UM90033 NLS FILE NAMING CONVENTIONS ENHANCEMENT
*        DMSMGC   A33398DS C1 XA1396  09/13/88   14:34:32
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A34362DS 203 UM04083 GENMSG CALCULATES INDEX PAST END OF PAGE.
*        DMSMGC   A34362DS A5 XA1191  11/03/88   13:03:06
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
*        DMSSP    MACLIB   S2 XA1390  10/21/88   10:43:25
  ⋮
*        DMSMGC   DMSTSP1  A1 XA1191  11/03/88   13:10:22
¢ESD ...
¢TXT ...
  ⋮
¢TXT ...
¢RLD ...
¢END ...
```

Figure 17. Example of a Text Deck – DMSMGC TXT04083

Text Shells

VMFREC creates a text shell on the DELTA disk for each PTF listed in the prolog of a text deck for which neither a text deck nor a text shell is available on the PUT or on the DELTA or LOCAL minidisks. The text shell contains:

1. A description of the APAR, including the APAR number, PUT level, PTF number, and content (from the AUX file)
2. Prerequisites and corequisites for this service, consisting of PREREQ, CO-REQ, and IF-REQ entries and agreeing with the requisites in the update file and update shell
3. A date and time stamp
4. The macro libraries used (optional).

The text shell does not include the executable text. VMFBLD will not accept a text shell as input. The shell is provided so that a search by PTF number will identify all the modules affected by the PTF.

If the filetype of an update file listed in a text shell begins with a percent sign (%), that update is listed in the control file and not in an auxiliary control file. Figure 18 on page 376 is an example of a text shell.

```
A33398DS 203 UM90033 NLS FILE NAMING CONVENTIONS ENHANCEMENT
*         DMSMGC   A33398DS C1 XA1396  09/13/88   14:34:32
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A34362DS 203 UM04083 GENMSG CALCULATES INDEX PAST END OF PAGE.
*         DMSMGC   A34362DS A5 XA1191  11/03/88   13:03:06
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
```

Figure 18. Example of a Text Shell – DMSMGC TXT04083

The PTF Parts List

The filename of the PTF (program temporary fix) parts list is the PTF number and the filetype is $PTFPART. It contains a list of the parts needed to install the PTF. This list should identify the appropriate text file even when the text file is not shipped on the PUT. The filemode of the PTF parts list is changed to filemode 5 after the PTF is applied.

The Apply List

The apply list contains a list of the PTFs to be applied to a product. The first line in the apply list is always a comment identifying the product and PUT level. An asterisk (*) indicates a comment and may be used to exclude a PTF. Figure 19 is an example of an apply list.

```
* APPLY LIST FOR PRODUCT 56643089 COR 0109
*VM Corrective Service Tape 0109 1X952 CSD  4817505 90/01/09 11:35:39.755240
UM12083 VM37944
UM12090 VM38482
UM12091 VM38575
UM12092 VM38430
UM12093 VM38247
UM12094 VM37999
UM12095 VM38821
UM12096 VM38676
```

Figure 19. Example of an Apply List – DMSXA $APPLIST

The PUT contains an apply list for each PUT level and one for all the levels on the PUT tape. The filename of the apply list on the service tape is the same as the filename of the control file specified in the product parameter file. (If you change the filename of the control file in the PPF, the filename of the apply list is automatically changed too.) The filetype is:

- For PUT service:
  - $APP*nnnn* for the apply list associated with a single PUT level, where *nnnn* is the PUT level
  - $APPALL for the cumulative apply list

- For COR service:
  - $APC*ymdd* for the apply list associated with a single COR tape, where *ymdd* is the date (last digit of the year, month numbered in hexadecimal, day)
  - $APCALL for the cumulative apply list.

VMFREC will copy the cumulative apply list to a file whose filetype is $APPLIST, and append any cumulative apply list it finds on the intermediate alternate disk(s). *fn* $APPLIST is the apply list used by the VMFAPPLY EXEC. If you want VMFAPPLY to use a different apply list, you must copy it with a filetype of $APPLIST. (Use the REPLACE option.) Copy it instead of renaming it so that you are sure to keep the apply list you used.

The Exclude List

The exclude list contains a list of the PTFs that are not to be applied to a product, even though they are listed in the apply list. Entries must be in the same order as in the apply list. The first line in the exclude list is always a comment identifying the product and PUT level. Figure 20 is an example of an exclude list.

```
* EXCLUDE LIST FOR PRODUCT 56643089 COR 9110
UM03601
UM90033
```

Figure 20. Example of an Exclude List – DMSXA $EXCLIST

The PUT contains an exclude list for each PUT level and one for all the levels on the PUT tape. The filename of the exclude list on the service tape is the same as the filename of the control file and apply list. The filetype is:

- For PUT service:
  - $EXP*nnnn* for the exclude list associated with a single PUT level, where *nnnn* is the PUT level
  - $EXPALL for the cumulative exclude list
- For COR service:
  - $EXC*ymdd* for the exclude list associated with a single COR tape, where *ymdd* is the date (last digit of the year, month numbered in hexadecimal, day)
  - $EXCALL for the cumulative exclude list.

VMFREC will copy the cumulative exclude list to a file whose filetype is $EXCLIST, and append any cumulative exclude list it finds on the intermediate alternate disk(s). *fn* $EXCLIST is the exclude list used by the VMFAPPLY EXEC. If you want VMFAPPLY to use a different exclude list, you must copy it with a filetype of $EXCLIST. (Use the REPLACE option.) Copy it instead of renaming it so that you are sure to keep the exclude list you used.

Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prereqisite for the second PTF.

If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs are applied. You receive an error message indicating that a PTF you wanted to exclude has been applied.

The Exception Log

The exception log is called $VMF*xxx* $ERRLOG, where *xxx* can be:

- REC for the receive exception log created by the VMFREC EXEC
- APP for the apply exception log created by the VMFAPPLY EXEC
- BLD for the build exception log created by the VMFBLD EXEC.

The exception log contains any error or informational messages issued by its corresponding EXEC and is written to the A-disk. If the exception log already exists when the EXEC is invoked, a date and time stamp is inserted to separate the earlier log entries from the new entries. You should browse the exception log with the VMFVIEW EXEC after VMFREC, VMFAPPLY, or VMFBLD finishes processing. Table 11 shows the different types of messages that may appear in an exception log.

Table 11. Exception Log Message Codes

Code	Explanation
ST:	A status message pertaining to the major function of the current process
BD:	Informational messages issued during the build process
CK:	A message that must be checked
WN:	A warning message
RO:	Messages pertaining to requisites outside the component
RQ:	Messages pertaining to requisites
MS:	Mismatched parts, such as AUX files and
SV:	A severe problem encountered AUX entries in the front of text decks

Note: All messages except status messages **must** be investigated. A warning message does not necessarily mean that anything is wrong, but **you cannot be sure until you check it out.**

Exception logs are cumulative. The most recent entries are at the **top.**

The Receive Exception Log

```
        ********************************************************************
        *****    RECEIVE  -  02/28/90     -     13:53:15  - run     *****
        ********************************************************************
        *****     Product:  56643089      Component:  CMS           *****
        ********************************************************************
ST:DMSREC1852I This is VOL01 of 01, level 9C22 COR TAPE.
WN:DMSREC1968W The Service Enhancements part set on DISK is at an
WN:            UNKNOWN level.
ST:DMSREC1967I The Service Enhancements part set in the TAPE header
ST:            files is at the 8902.2 level.
ST:DMSREC1806I The current SERVICE DISKMAP file contains
ST:            the map of the mounted tape.
ST:DMSREC1804I Receiving service for component CMS of product 56643089
ST:DMSREC1940I Merge processing is not required for component CMS.
ST:DMSREC1851I Processing AXLIST  with the part handler VMFRCAXL EXEC.
ST:DMSREC1851I Processing PARTLST  with the part handler VMFRCCOM EXEC.
ST:DMSREC1851I Processing UPDT  with the part handler VMFRCUPD EXEC.
ST:DMSREC1851I Processing TEXT  with the part handler VMFRCTXT EXEC.
ST:DMSREC1851I Processing MACAUX  with the part handler VMFRCAUX EXEC.
ST:DMSREC1851I Processing MACUPDT  with the part handler VMFRCUPD EXEC.
ST:DMSREC1851I Processing SOURCE  with the part handler VMFRCCOM EXEC.
ST:DMSREC1851I Processing EXEC  with the part handler VMFRCCOM EXEC.
ST:DMSREC1851I Processing HELP  with the part handler VMFRCCOM EXEC.
ST:DMSREC1851I Processing HELP  with the part handler VMFRCUPP EXEC.
ST:DMSREC1850I The processing for RESV by the VMFRCCOM EXEC was bypassed.
        ********************************************************************
        *****    RECEIVE  -  02/28/90     -     13:49:59  - run     *****
        ********************************************************************
        *****                 INFO option                            *****
        ********************************************************************
ST:DMSREC1852I This is VOL01 of 01, level 9C22 COR TAPE.
WN:DMSREC1968W The Service Enhancements part set on DISK is at an
WN:            UNKNOWN level.
ST:DMSREC1967I The Service Enhancements part set in the TAPE header
ST:            files is at the 8902.2 level.
```

Figure 21. Example of a Receive Exception Log – $VMFREC $ERRLOG

The Apply Exception Log

```
        ****************************************************************
        *****    APPLY    -  02/28/90     -     14:01:46  - run     *****
        ****************************************************************
        *****     Product:  56643089      Component:  CMS           *****
        ****************************************************************
MNT191  191   A    R/W
MNT5E5  5E5   B    R/W
MNT692  692   D    R/W
MNT391  391   E    R/W
MNT392  392   F    R/W
MNT593  593   G    R/W
MNT691  691   H    R/W
MNT293  293   I    R/W
MNT395  395   J    R/W
MNT193  193   K    R/W
MNT393  393   L    R/W
MNT490  490   M    R/W
MNT49D  49D   N    R/W
MNT501  501   O    R/W
MNT49C  49C   P    R/W
MNT295  295   Q    R/W
MNT594  594   R    R/W
MNT190  190   S    R/O
MNT591  591   T    R/W
MNT294  294   U    R/W
MNT194  194   V    R/W
MNT19E  19E   Y/S  R/O
ST:DMSAPP1853I Processing PTF UM12083
ST:DMSAPP1851I Processing DMSDSK A37944DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing DMSDSK TXT12083 with the part handler VMFAPTXT EXEC.
BD:DMSAPX1858I DMSDSK TEXT has been serviced.
ST:DMSAPP1853I Processing PTF UM12090
ST:DMSAPP1851I Processing DMSLBM A38482DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing DMSLBM TXT12090 with the part handler VMFAPTXT EXEC.
BD:DMSAPX1858I DMSLBM TEXT has been serviced.
ST:DMSAPP1853I Processing PTF UM12091
ST:DMSAPP1851I Processing HELPXED AUXXA with the part handler VMFAPAUX EXEC.
BD:DMSWAU1858I HELPXED AUXXA has been serviced.
ST:DMSAPP1851I Processing HELPXED A38575DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing HELPXED XED12091 with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing HELPXED XEDIT with the part handler VMFAPCOM EXEC.
ST:DMSAPP1853I Processing PTF UM12092
ST:DMSAPP1851I Processing DMSCIT A38430DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing DMSCIT TXT12092 with the part handler VMFAPTXT EXEC.
```

Figure 22 (Part 1 of 2). Example of an Apply Exception Log—$VMFAPP $ERRLOG

```
BD:DMSAPX1858I CMSLOAD build list contains files that
BD:            have been serviced.
ST:DMSAPP1853I Processing PTF UM12093
ST:DMSAPP1851I Processing DMSDSK A38247DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing DMSDSK TXT12093 with the part handler VMFAPTXT EXEC.
ST:DMSAPP1853I Processing PTF UM12094
ST:DMSAPP1851I Processing DMSCRD A37999DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing DMSCRD TXT12094 with the part handler VMFAPTXT EXEC.
ST:DMSAPP1853I Processing PTF UM12095
ST:DMSAPP1851I Processing VMFAPPLY AUXXA with the part handler VMFAPAUX EXEC.
BD:DMSWAU1858I VMFAPPLY AUXXA has been serviced.
ST:DMSAPP1851I Processing VMFAPPLY A38821DS with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing VMFAPPLY EXC12095 with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing VMFAPPLY EXEC with the part handler VMFAPCOM EXEC.
ST:DMSAPP1853I Processing PTF UM12096
ST:DMSAPP1851I Processing GENMOD HCM12096 with the part handler VMFAPCOM EXEC.
ST:DMSAPP1851I Processing GENMOD HELPCMS with the part handler VMFAPCOM EXEC.
```

Figure 22 (Part 2 of 2). Example of an Apply Exception Log – $VMFAPP $ERRLOG

The Build Exception Log

```
        ********************************************************************
        *****    BUILD    -  02/28/90     -     14:15:29  - run      *****
        ********************************************************************
        *****     Product:  56643089      Component:  CMS            *****
        ********************************************************************
ST:DMSBLD1851I Processing CMSLOAD  with the part handler VMFBDNUC EXEC.
ST:DMSBLD1851I Processing CPYSYSCM  with the part handler VMFBDCPY EXEC.
ST:DMSBLD1851I Processing CPYBTHCM  with the part handler VMFBDCPY EXEC.
```

Figure 23. Example of a Build Exception Log – $VMFBLD $ERRLOG

The Receive History Log

The receive history log has a filename of $VMFREC and a filetype of $HISTORY. The receive history log contains a list of all CMS files received from the PUT when VMFPLC2 is invoked by VMFREC for any component of VM/XA SP, and shows the date and time when each file is received.

Note: Files loaded from tape using any of the INFO options of VMFREC EXEC do not appear in the receive history log. Therefore, the various tape documents, the service disk map, and the service EXECs will not appear in the receive history log.

The receive history log is cumulative. It differs from the exception logs in that the most recent entries are at the **bottom**.

```
*********************************************************
Receiving: 56643089 - CMS 15 Jan 1990 13:59:43
*********************************************************
 LOADING.....
 DMSXA    $APCALL  G1
 DMSXA    $APC9C22 G1
 DMSXA    $EXCALL  G1
 DMSXA    $EXC9C22 G1
 END-OF-FILE OR END-OF-TAPE
 LOADING...
 UM12603  $PTFPART U1
 END-OF-FILE OR END-OF-TAPE
 LOADING...
 DMSGLB   A39617DS U1

  :
```

Figure 24. Example of a VMFREC History File – $VMFREC $HISTORY

The Load List

The load list is an ordered listing of the CP, CMS, or GCS modules. The ordering (top to bottom) determines which modules come first on the system residence device and which modules come into storage upon an initial program load.

The CP Load List

For the CP load list, the order is determined by the HCPMDLAT MACRO, which is invoked by the HCPLDL ASSEMBLE module. The load list is created when HCPLDL is assembled. CPLOAD EXEC is shipped on the PUT service tape, but is not shipped on the COR tape. Therefore UTILITY must be used to generate the CPLOAD EXEC only if COR service is applied or if local modifications are made.

The file created by assembling HCPLDL is called HCPLDL TEXT or HCPLDL TXT*xxxxx*. If it is called HCPLDL TXT*nnnnn*, you must rename it HCPLDL TEXT. You must then use the UTILITY EXEC to convert HCPLDL TEXT to CPLOAD EXEC.

The modules fall into these general categories:

- Resident modules. These modules are nonpageable.
- Pageable modules loaded at initialization. These modules may not be paged until system initialization has finished. Thereafter, the modules can be paged out.
- Other pageable modules. These modules, along with the modules that are resident only during initialization, comprise the part of the system that CP pages in and out of real storage.

The CP load list must have fixed-length 80-byte records.

Do not change the CP load list unless it is absolutely necessary.

The CMS Load List

The CMS load list is called CMSLOAD EXEC. Do not change anything in the CMS load list except the SLC (Set Location Counter) cards, which point to files that determine where the modules in the CMS resident nucleus are loaded.

The GCS Load List

The GCS load list is called GCSLOAD EXEC. Do not change anything in the GCS load list.

The Temporary Load List

The temporary load list is a file generated by VMFBLD. It includes all of the text decks in the original load list (CPLOAD EXEC for CP, CMSLOAD EXEC for CMS, GCSLOAD EXEC for GCS, etc.) as well as the filetype of the "latest" version of the text deck (latest being determined by the CNTRL and AUX file structure). HCPLDR MODULE generates the CP, CMS, and GCS nuclei using the temporary load list for each component.

```
&TRACE OFF
&1 &2 &3 HCPLDR LOADER
&1 &2 &3 HCPSYS TXT12345
&1 &2 &3 HCPRIO TXT22333
&1 &2 &3 HCPDDI TXT98765
  ⋮
&1 &2 &3 HCPMM3 TXT22333
```

Figure 25. Example of a Temporary Load List – $$$TLL$ EXEC

The Load Map

The load map has the same filename as the control file and a filetype of LOADMAP. It contains the map of CSECT external symbol resolution and service level information from the text decks, which are self-documenting. It is written to the A-disk if you specify the DISK option for the VMFBLD EXEC. Otherwise, it is spooled to the printer.

If the filetype of an update file listed in the load map begins with a percent sign (%), that update is listed in the control file and not in an auxiliary control file.

Restart Indicator Files

| The VMFREC EXEC stores one or two restart indicator files on the alternate apply disk. Do not erase the restart indicator files. If you need to restart VMFREC, they determine how VMFREC will proceed.

The restart indicator files are called:

- For PUT service:

  prodid $MRP*nnnn*

- For COR service:

  prodid $MRC*ymdd*
  $COR *ymdd*

  Where:

  prodid
  is the identifier of the product for which service is being received (for example, 56643089).

  nnnn
  is the PUT level.

ymdd
is the date (last digit of year, month numbered in hexadecimal, day).

If the $MR*xxxxx* file is present, the files for PUT level *nnnn* or COR service dated *ymdd* have been received and VMFREC has successfully completed. The MERGE disks will be merged from alternate to current when VMFREC is rerun for the same product or component. The components for which service has been received are listed in the $MR*xxxxx* file.

$COR *ymdd* contains the first record of the COR descriptor file, which contains the time when the tape was built. If this file is present for a tape you want to receive, it indicates that VMFREC has already been run for this tape.

The Product Parameter File

You must have a product parameter file in order to update products supported by the new service EXECs, including VM/XA System Product. The filename of the product parameter file is the product ID—for example, 56643089—and the filetype is $PPF.

The product parameter file consists of a product area, one or more component areas, and one or more override areas.

- The product area (introduced by the :COMPLST. tag) lists the components of the product.
- Each component area (introduced by a *:compname.* tag) has three sections:
  - The first section (introduced by the :CNTRLOP. tag) contains control options such as the control file name, the system level and version, and the national language.
  - The second section (introduced by the :MDA. tag) contains minidisk assignments:
    - The TASK disks, where you can isolate the latest level of service EXECs
    - The LOCAL disks, where you keep local update files
    - The APPLY disk, where the auxiliary control structure will be built
    - The DELTA disks, where you will load the update files from the program update tape (PUT)
    - The BASE disks, holding the component's source and object code
    - The BUILD disks, on which you will build your new system
    - The system's minidisks
  - The third section (introduced by the :PRTFNC. tag) lists the EXECs needed to service each part of the component. There are three functional sections:
    - Receive (introduced by the :RECSER. tag)
    - Apply (introduced by the :APP. tag)
    - Build (introduced by the :BLD. tag)
- Each override area (introduced by an *:overname.* tag) contains changes that can be made to the corresponding component area when the user wishes. The override area permits you to build several different systems with a single product parameter file.

A Sample Product Parameter File

The listing below is a sample product parameter file. The tags are explained in "The Tags in the Product Parameter File" on page 395.

```
*********************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1989
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083
*********************************************************************
*====================================================================
*            Product Parameter File for VM/XA SP Release 2.1
*====================================================================
*
* The following APAR's have been applied to this file:
*
*  APAR      Date       Description
*  =======   ========   ===========================================
*
*
*====================================================================
*            NOTE: All tags must be in uppercase.
*====================================================================

*====================================================================
* Start of Product Header - List of Components in VM/XA SP Release 2.1
*====================================================================
:COMPLST.      CP    CMS    DV    GCS
:OVERLSTP.
:OVERLST.   CORCP CORCMS CORDV CORGCS
*====================================================================
* End of Product Header
*====================================================================

*====================================================================
:CP.                          * Start of Parameters for CP
*====================================================================
:CNTRLOP.                     * Section one - Control Options
**                            * DO NOT DELETE THESE CONTROL TAGS
* TAG      VALUE(S)           * DESCRIPTION
**
:CNTRL.    HCPXA              * Control Filename
:UPDTID.   AUXXA              * Update level identifier for building AUX
:SLVI.     A/HP A/PP          * System level and version indicators
:NLS.      AMENG              * System Language
:USEREXIT. VMFUECP            * User exit EXEC.  Called before initial
**                            * accesses and after final access retore.
**                            * 2 paramters passed will indicate calling
**                            * EXEC and SET-UP or CLEAN-UP.
:MERGE.    DELTA1 APPLY       * Disk strings to be merged at Receive
:LTO.      NO                 * Erase lower level TEXT decks at Receive
:CKTXT.    YES                * Check for full text at Apply
:CKUPD.    NO                 * Check update fm regardless of $PTFPART fm
:CKAUX.    YES                * Check composite AUX at Build
:CKREQ.    YES                * Check Requisites at Build
*====================================================================
:MDA.                         * Section two - Mini-Disk Assignments
**                            * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS           * DESCRIPTION
**
TASK       5E5                * Build disks accessed before rest of dbase
LOCAL1     295                * Disks for local service
*LOCALn                       * You may define additional LOCAL disks
APPLY      492 291 292        * Control structure built by service execs
*APPLYn                       * You may define additional APPLY disks
```

```
DELTA1     594 591 294       * CP service from TAPE
*DELTAn                      * You may define additional DELTA disks
BASE1      194               * CP text,loadlist,control file,maclibs
BASE2      394               * CP source,macro defs,and copy files
*BASEn                       * You may define additional BASE disks
*BUILD1    423               * New SYSRES for CP Nucleus Build
BUILD2     49D               * New CMS system disk for AMENG HELP files
BUILD3     49C               * New CMS system disk for UCENG HELP files
BUILD4     490               * New CMS system disk for nucleus etc.
*BUILDn                      * You may define additional BUILD disks
SYSTEM     191               * Disks reqd for your system environment
*=============================================================================
:PRTFNC.                     * Section three - Part type/Function List
*=============================================================================
:RECSER.                     * Receive (Service)
**
* TAPEFILE EXEC     TARGET   * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1   * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1   * PTF Parts Lists
  UPDT     VMFRCUPD DELTA1   * Update files for CP ASSEMBLE source
  TEXT     VMFRCTXT DELTA1   * Text replacement modules for CP
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
  MACAUX   VMFRCAUX DELTA1   * AUX files for updates to non-ASSEM parts
  MACUPDT  VMFRCUPD DELTA1   * Update files for non-ASSEMBLE parts
  SOURCE   VMFRCCOM DELTA1   * New Source (assemble,macro,copy,$exec...)
  MACLIB   VMFRCCOM DELTA1   * Maclibs are not replaced on the base
  EXEC     VMFRCCOM DELTA1   * Replacement parts and regenerated
**                           * parts other than maclibs, text, modules
  HELP     VMFRCCOM BUILD2   * AMENG HELP files go directly to the
**                           * AMENG help Build disk
  HELP     VMFRCUPP BUILD3   * UCENG HELP files go directly to the
**                           * UCENG help Build disk
  RESV    *VMFRCCOM DELTA1   * Reserved parts
  CPMOD    VMFRCCOM BUILD4   * CP modules targeted for system disk
  PTFS     VMFRCCOM DELTA1   * PTF numbered parts other than TXT
*=============================================================================
:APP.                        * Apply Function
**
* PARTTYPE EXEC     RENAME   * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT     * Create AUX and rename appropriate update
**                           * files to fm5 for names in loadlist.
**                           * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC * For French NLS support
  TGR*     VMFAPNLS TXTGER   * For German NLS support
  AUXXA    VMFAPAUX          * For MACRO and EXEC AUX files
  AXA*     VMFAPAUX AUXXA    * For MACRO and EXEC AUX files
  EXEC     VMFAPCOM          *
  MACRO    VMFAPCOM          *
** .       VMFAPCOM          * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM          * If a part type is not listed
** .       VMFAPCOM          * VMFAPCOM is called by default
*=============================================================================
:BLD.                        * Build Function
```

```
**
* BUILDLST EXEC     TARGET   * DESCRIPTION
**
  CPLOAD   VMFBDNUC *BUILD1  * Create CP nucleus based on loadlist
**                           * Target disk symbol agrees with
**                           * disk address in HCPSYS ASSEMBLE
**                           * on the SYSRES macro. Defaults to
**                           * what is in HCPSYS ASSEMBLE if no
**                           * target disk is specified.
**
  CPYSYSCP VMFBDCPY BUILD4   * Copy TEXT decks that do not become
**                           * part of a MODULE to the system disk
**
  CPYBTHCP VMFBDCPY DELTA1   * Copy TEXT decks used by UTILITY and
**                           * listed in the load list, CPLOAD EXEC,
**                           * to the DELTA disk. VMFAPPLY only
**                           * copies the ones not listed in the
**                           * load list.
**
* CPYUTLCP VMFBDCPY DELTA1   * Copy TEXT decks used by UTILITY which
**                           * are not listed in the load list,
**                           * CPLOAD EXEC. This list would be used
**                           * if you have local modifications and
**                           * you choose not to run VMFAPPLY. All
**                           * TEXT decks for which you have received
**                           * service and/or have local mods would
**                           * be copied as a result. You need to remove
**                           * the asterisk before CPYUTLCP before
**                           * running VMFBLD.
**                           * Use UTILITY to build CP Utilities
*============================================================================
:END.                        * End of Parameters for CP
*============================================================================

*============================================================================
:CORCP.  CP                  * Start of CP Corrective Service Overrides
*============================================================================
:MERGE.    APPLYX DELTAX     * Merge dummy APPLY and DELTA strings
:MDA.      UPDATE
APPLYX     492 291           * Define dummy APPLY string
DELTAX     594 591           * Define dummy DELTA string
**                           * During corrective service merges are per-
**                           * formed based on these dummy strings.  The
**                           * intermediate disks act as collectors for
**                           * corrective service until the next PUT.
*============================================================================
:END.                        * End of CP Corrective Service Overrides
*============================================================================

*============================================================================
:CMS.                        * Start of Parameters for CMS
*============================================================================
:CNTRLOP.                    * Section one - Control Options
**                           * DO NOT DELETE THESE CONTROL TAGS
* TAG       VALUE(S)        * DESCRIPTION
**
:CNTRL.    DMSXA             * Control Filename
:UPDTID.   AUXXA             * Update level identifier for building AUX
:SLVI.     A/DS E/DS         * System level and version indicators
:NLS.      AMENG             * System Language
```

```
:USEREXIT. VMFUECMS          * User exit EXEC.  Called before initial
**                           * accesses and after final access retore.
**                           * 2 paramters passed will indicate calling
**                           * EXEC and SET-UP or CLEAN-UP.
:MERGE.    DELTA1 APPLY      * Disk strings to be merged at Receive
:LTO.      NO                * Erase lower level TEXT decks at Receive
:CKTXT.    YES               * Check for full text at Apply
:CKUPD.    NO                * Check update fm regardless of $PTFPART fm
:CKAUX.    YES               * Check composite AUX at Build
:CKREQ.    YES               * Check Requisites at Build
*=============================================================================
:MDA.                        * Section two - Mini-Disk Assignments
**                           * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS         * DESCRIPTION
**
TASK       5E5               * Build disks accessed before rest of dbase
LOCAL1     395               * Disks for local service
*LOCALn                      * You may define additional LOCAL disks
APPLY      692 391 392       * Control structure built by service execs
*APPLYn                      * You may define additional APPLY disks
DELTA1     593 691 293       * CMS service from TAPE
*DELTAn                      * You may define additional DELTA disks
BASE1      193               * CMS text,loadlist,control file,maclibs
BASE2      393               * CMS source and macro definitions
*BASEn                       * You may define additional BASE disks
BUILD1     490               * New CMS system disk for nucleus etc.
BUILD2     49D               * New CMS system disk for AMENG HELP files
BUILD3     501               * New CMS system disk for EREP files
BUILD4     49C               * New CMS system disk for UCENG HELP files
*BUILDn                      * You may define additional BUILD disks
SYSTEM     191 295 594 591 294 194
**                           * Disks reqd for your system environment.
**                           * CP database needed for MACLIBs.  If
**                           * latest MACLIBs are on a build disk, only
**                           * that disk must appear here.
*=============================================================================
:PRTFNC.                     * Section three - Part type/Function List
*=============================================================================
:RECSER.                     * Receive (Service)
**
* TAPEFILE EXEC     TARGET   * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1   * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1   * PTF Parts Lists
  UPDT     VMFRCUPD DELTA1   * Update files for CMS ASSEMBLE source
  TEXT     VMFRCTXT DELTA1   * Text replacement modules for CMS
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
  MACAUX   VMFRCAUX DELTA1   * AUX files for updates to non-ASSEM parts
  MACUPDT  VMFRCUPD DELTA1   * Update files for non-ASSEMBLE parts
  SOURCE   VMFRCCOM DELTA1   * New Source (assemble,macro,copy,$exec...)
  MACLIB   VMFRCCOM BUILD1   * Maclibs go directly to the Build disk
  EXEC     VMFRCCOM DELTA1   * Replacement parts and regenerated
**                           * parts other than maclibs, text, modules
  MODULE   VMFRCCOM BUILD1   * CMS/IOCP modules go directly to the
**                           * Build disk
  HELP     VMFRCCOM BUILD2   * AMENG HELP files go directly to the
**                           * AMENG help Build disk
  HELP     VMFRCUPP BUILD4   * UCENG HELP files go directly to the
**                           * UCENG help Build disk
```

```
  IOCP     VMFRCTXT DELTA1   * IOCP files go to the DELTA1 disk
  RESV    *VMFRCCOM DELTA1   * Reserved parts
  PTFS     VMFRCCOM DELTA1   * ptf numbered parts other than TXT
*=============================================================================
:APP.                       * Apply Function
**
* PARTTYPE EXEC     RENAME   * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT     * Create AUX and rename appropriate update
**                            * files to fm5 for names in loadlist.
**                            * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC * For French NLS support
  TGR*     VMFAPNLS TXTGER   * For German NLS support
  AUXXA    VMFAPAUX          * For EXEC AUX files
  AXA*     VMFAPAUX AUXXA    * For EXEC AUX files
  AUXMXA   VMFAPAUX          * For MACRO AUX files
  AXM*     VMFAPAUX AUXMXA   * For MACRO AUX files
  EXEC     VMFAPCOM          *
  MACRO    VMFAPCOM          *
** .       VMFAPCOM          * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM          * If a part type is not listed
** .       VMFAPCOM          * VMFAPCOM is called by default
*=============================================================================
:BLD.                       * Build Function
**
* BUILDLST EXEC     TARGET   * DESCRIPTION
**
  CMSLOAD  VMFBDNUC BUILD1   * Create CMS nucleus based on loadlist
**
  CPYSYSCM VMFBDCPY BUILD1   * Copy TEXT decks that do not become
**                            * part of a MODULE to the system disk
**
  CPYBTHCM VMFBDCPY DELTA1   * Copy TEXT decks used by CMSGEND and
**                            * listed in the load list, CMSLOAD EXEC,
**                            * to the DELTA disk. VMFAPPLY only
**                            * copies the ones not listed in the load
**                            * list.
**
* CPYGNDCM VMFBDCPY DELTA1   * Copy TEXT decks used by CMSGEND which
**                            * are not listed in the load list,
**                            * CMSLOAD EXEC. This list would be used
**                            * if you have local modifications and
**                            * you choose not to run VMFAPPLY. All
**                            * TEXT decks for which you have received
**                            * service and/or have local mods would
**                            * be copied as a result. You need to remove
**                            * the asterisk before CPYGNDCM before
**                            * running VMFBLD.
**                            * Use CMSGEND to build CMS modules
*=============================================================================
:END.                       * End of Parameters for CMS
*=============================================================================

*=============================================================================
:CORCMS. CMS                * Start of CMS Corrective Service Overrides
*=============================================================================
```

```
:MERGE.    APPLYX DELTAX     * Merge dummy APPLY and DELTA strings
:MDA.      UPDATE
APPLYX     692 391           * Define dummy APPLY string
DELTAX     593 691           * Define dummy DELTA string
**                           * During corrective service merges are per-
**                           * formed based on these dummy strings.  The
**                           * intermediate disks act as collectors for
**                           * corrective service until the next PUT.
*============================================================================
:END.                        * End of CMS Corrective Service Overrides
*============================================================================

*============================================================================
:DV.                         * Start of Parameters for DV
*============================================================================
:CNTRLOP.                    * Section one - Control Options
**                           * DO NOT DELETE THESE CONTROL TAGS
* TAG       VALUE(S)         * DESCRIPTION
**
:CNTRL.    HCSXA             * Control Filename
:UPDTID.   AUXXA             * Update level identifier for building AUX
:SLVI.     A/HP A/PP         * System level and version indicators
:NLS.      AMENG             * System Language
:USEREXIT. VMFUEDV           * User exit EXEC.  Called before initial
**                           * accesses and after final access retore.
**                           * 2 paramters passed will indicate calling
**                           * EXEC and SET-UP or CLEAN-UP.
:MERGE.    DELTA1 APPLY      * Disk strings to be merged at Receive
:LTO.      NO                * Erase lower level TEXT decks at Receive
:CKTXT.    YES               * Check for full text at Apply
:CKUPD.    NO                * Check update fm regardless of $PTFPART fm
:CKAUX.    YES               * Check composite AUX at Build
:CKREQ.    YES               * Check Requisites at Build
*============================================================================
:MDA.                        * Section two - Mini-Disk Assignments
**                           * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS         * DESCRIPTION
**
TASK       5E5               * Build disks accessed before rest of dbase
LOCAL1     395               * Disks for local service
*LOCALn                      * You may define additional LOCAL disks
APPLY      692 391 392       * Control structure built by service execs
*APPLYn                      * You may define additional APPLY disks
DELTA1     593 691 293       * DV service from TAPE
*DELTAn                      * You may define additional DELTA disks
BASE1      193               * DV text,loadlist,control file,maclibs
*BASEn                       * You may define additional BASE disks
BUILD1     490               * New system disk for CMS and Dump Viewer
BUILD2     49D               * New CMS system disk for AMENG HELP files
BUILD4     49C               * New CMS system disk for UCENG HELP files
*BUILDn                      * You may define additional BUILD disks
SYSTEM     191 295 594 591 294 194
**                           * Disks reqd for your system environment.
**                           * CP database needed for MACLIBs.  If
**                           * latest MACLIBs are on a build disk, only
**                           * that disk must appear here.
*============================================================================
:PRTFNC.                     * Section three - Part type/Function List
*============================================================================
:RECSER.                     * Receive (Service)
```

```
**
* TAPEFILE EXEC     TARGET   * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1   * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1   * PTF Parts Lists
  UPDT     VMFRCUPD DELTA1   * Update files for HCSTBL ASSEMBLE source
  DVFTEXT  VMFRCTXT DELTA1   * Text replacement modules for DV
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
  EXECAUX  VMFRCAUX DELTA1   * AUX files for updates to non-ASSEM parts
  EXECUPD  VMFRCUPD DELTA1   * XEDIT updates to DELTA disk
**                           * (DVFXEDIT $XEDIT and FINDUSER $XEDIT)
  SOURCE   VMFRCCOM DELTA1   * New Source (assemble,macro,copy,$exec...)
  DVFEXEC  VMFRCCOM BUILD1   * DUMP VIEWING EXECs go directly
**                           * to the Build disk
**                           * includes CPBLOCK CBMAP and HCSCP1 TABLE
  DVFMODS  VMFRCCOM BUILD1   * DUMP VIEWING modules go directly
**                           * to the Build disk
  HELP     VMFRCCOM BUILD2   * AMENG HELP files go directly to the
**                           * AMENG help Build disk
  HELP     VMFRCUPP BUILD4   * UCENG HELP files go directly to the
**                           * UCENG help Build disk
  RESV    *VMFRCCOM DELTA1   * Reserved parts
  PTFS     VMFRCCOM DELTA1   * ptf numbered parts other than TXT
*=============================================================================
:APP.                        * Apply Function
**
* PARTTYPE EXEC     RENAME   * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT     * Create AUX and rename appropriate update
**                           * files to fm5 for names in loadlist.
**                           * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC * For French NLS support
  TGR*     VMFAPNLS TXTGER   * For German NLS support
  AUXXA    VMFAPAUX          * For EXEC AUX files
  AXA*     VMFAPAUX AUXXA    * For EXEC AUX files
  EXEC     VMFAPCOM          *
** .       VMFAPCOM          * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM          * If a part type is not listed
** .       VMFAPCOM          * VMFAPCOM is called by default
*=============================================================================
:BLD.                        * Build Function
**
* BUILDLST EXEC     TARGET   * DESCRIPTION
**
  CPYSYSDV VMFBDCPY BUILD1   * Copy DV Interface Files to System disk
**
**                           * Use UTILITY DVFGEND to build Dump Viewer
*=============================================================================
:END.                        * End of Parameters for DV
*=============================================================================

*=============================================================================
:CORDV.  DV                  * Start of DV Corrective Service Overrides
*=============================================================================
:MERGE.    APPLYX DELTAX     * Merge dummy APPLY and DELTA strings
```

```
:MDA.       UPDATE
APPLYX      692 391          * Define dummy APPLY string
DELTAX      593 691          * Define dummy DELTA string
**                           * During corrective service merges are per-
**                           * formed based on these dummy strings.  The
**                           * intermediate disks act as collectors for
**                           * corrective service until the next PUT.
*============================================================================
:END.                        * End of DV Corrective Service Overrides
*============================================================================

*============================================================================
:GCS.                        * Start of Parameters for GCS
*============================================================================
:CNTRLOP.                    * Section one - Control Options
**                           * DO NOT DELETE THESE CONTROL TAGS
* TAG       VALUE(S)         * DESCRIPTION
**
:CNTRL.     CSIXA            * Control Filename
:UPDTID.    AUXXA            * Update level identifier for building AUX
:SLVI.      A/CI             * System level and version indicators
:NLS.       AMENG            * System Language
:USEREXIT.  VMFUECMS         * User exit EXEC.  Called before initial
**                           * accesses and after final access retore.
**                           * 2 paramters passed will indicate calling
**                           * EXEC and SET-UP or CLEAN-UP.
:MERGE.     DELTA1 APPLY     * Disk strings to be merged at Receive
:LTO.       NO               * Erase lower level TEXT decks at Receive
:CKTXT.     YES              * Check for full text at Apply
:CKUPD.     NO               * Check update fm regardless of $PTFPART fm
:CKAUX.     YES              * Check composite AUX at Build
:CKREQ.     YES              * Check Requisites at Build
*============================================================================
:MDA.                        * Section two - Mini-Disk Assignments
**                           * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS         * DESCRIPTION
**
TASK        5E5              * Build disks accessed before rest of dbase
LOCAL1      495              * Disks for local service
*LOCALn                      * You may define additional LOCAL disks
APPLY       892 491 592      * Control structure built by service execs
*APPLYn                      * You may define additional APPLY disks
DELTA1      896 791 596      * GCS service from TAPE
*DELTAn                      * You may define additional DELTA disks
BASE1       595              * GCS text,loadlist,control file,maclibs
**                           * Nucleus, LOADLIB, EXECs
*BASE2                       * GCS source files ASSEMBLE,$EXEC,$XEDIT
*BASEn                       * You may define additional BASE disks
BUILD1      895              * New GCS system disk for nucleus etc.
BUILD2      89E              * New GCS system disk extension
BUILD3      49D              * New CMS system disk for AMENG HELP files
BUILD4      49C              * New CMS system disk for UCENG HELP files
BUILD5      490              * New CMS system disk for GCS Interface
*BUILDn                      * You may define additional BUILD disks
SYSTEM      191              * Disks reqd for your system environment.
*============================================================================
:PRTFNC.                     * Section three - Part type/Function List
*============================================================================
:RECSER.                     * Receive (Service)
**
```

```
* TAPEFILE EXEC     TARGET   * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1   * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1   * PTF Parts Lists
  TEXT     VMFRCTXT DELTA1   * Text replacement modules for GCS
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
  EXECAUX  VMFRCAUX DELTA1   * AUX files for updates to non-ASSEM parts
  EXECUPD  VMFRCUPD DELTA1   * Update files for non-ASSEMBLE parts
  SOURCE   VMFRCCOM DELTA1   * NEW SOURCE FILES GO ON DELTA1
  MACLIB   VMFRCCOM BUILD1   * Maclibs go directly to the Build disk
  EXEC     VMFRCCOM DELTA1   * Replacement parts and regenerated
**                           * parts other than maclibs, text, modules
  MODULE   VMFRCCOM BUILD1   * GCS modules go directly to the Build disk
  HELP     VMFRCCOM BUILD3   * AMENG HELP files go directly to the
**                           * AMENG help Build disk
  HELP     VMFRCUPP BUILD4   * UCENG HELP files go directly to the
**                           * UCENG help Build disk
  RESV    *VMFRCCOM DELTA1   * Reserved parts
  PTFS     VMFRCCOM DELTA1   * PTF numbered parts other than TXT
  GCSCMS   VMFRCTXT DELTA1   * Text replacement GCS interface files.
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
*============================================================================
:APP.                        * Apply Function
**
* PARTTYPE EXEC     RENAME   * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT     * Create AUX and rename appropriate update
**                           * files to fm5 for names in loadlist.
**                           * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC * For French NLS support
  TGR*     VMFAPNLS TXTGER   * For German NLS support
  AUXXA    VMFAPAUX          * For MACRO and EXEC AUX files
  AXA*     VMFAPAUX AUXXA    * For MACRO and EXEC AUX files
  EXEC     VMFAPCOM          *
  MACRO    VMFAPCOM          *
** .       VMFAPCOM          * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM          * If a part type is not listed
** .       VMFAPCOM          * VMFAPCOM is called by default
*============================================================================
:BLD.                        * Build Function
**
* BUILDLST EXEC     TARGET   * DESCRIPTION
**
  GCSLOAD  VMFBDNUC BUILD1   * Create GCS NUCLEUS based on LOADLIST
**                           * Use GROUP EXEC to build GCS Macilbs
**
  CPYGCS   VMFBDCPY BUILD5   * Rename/Copy GCS Interface Files, needed
**                           * for DUMPSCAN formatters, these are
**                           * TEXT decks.
**
* CPYOTHGC VMFBDCPY DELTA1   * PMX modules or TEXT decks, message
**                           * exchangers, loaded in SNA program
**                           * products, that do not become part of
**                           * the GCS nucleus. If you choose not
```

```
**                          * to run VMFAPPLY, this list would
**                          * copy these text decks to the
**                          * DELTA disk during the VMFBLD process.
*=========================================================================
:END.                       * End of Parameters for GCS
*=========================================================================

*=========================================================================
:CORGCS. GCS                * Start of GCS Corrective Service Overrides
*=========================================================================
:MERGE.    APPLYX DELTAX    * Merge dummy APPLY and DELTA strings
:MDA.      UPDATE
APPLYX     892 491          * Define dummy APPLY string
DELTAX     896 791          * Define dummy DELTA string
**                          * During corrective service merges are per-
**                          * formed based on these dummy strings.  The
**                          * intermediate disks act as collectors for
**                          * corrective service until the next PUT.
*=========================================================================
:END.                       * End of GCS Corrective Service Overrides
*=========================================================================
```

Warning: The disk addresses in the SPLOAD PROFILE, the user directory, 56643089 $PPF, and the SETUP EXEC must all match the default addresses the ITASK EXEC uses. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

The Tags in the Product Parameter File

Product Area

:COMPLST. *compname1 compname2 ... compnamen comment*

:COMPLST.
is a list of the components in the product. It can be specified only in the product area of the base product parameter file (where it is required), not in an override file.

compname
is an unofficial nickname, in uppercase, for the component. The first component of the product on the PUT gets the first compname in the list, the second component the second compname and so on.

comment
is any string prefixed with an asterisk. A comment can appear after any of the tags or on a line by itself anywhere in the product parameter file.

:OVERLSTP. *overname1 overname2 ... overnamen*
is a list of override names that you want VMFOVER to prompt you with, in addition to the list of component names following the :COMPLST. tag, if VMFOVER is invoked without a component being specified. If you do not want VMFOVER to prompt you with any override names, specify :OVERLSTP. followed by an empty list.

If you do not specify the :OVERLSTP. tag, the names following the :OVERLST. tag are used for prompting. You do not have to list the override names twice. You will be prompted only once for any name listed on both tags.

The :OVERLSTP. tag is required in the product area of the base product parameter file and permitted in the product area of a separate override file. It is not allowed in the component override section of either file.

:OVERLSTP. must always be placed between the :COMPLST. tag and the :OVERLST. tag in the PPF.

:OVERLST. *overname1 overname2 ... overnamen*
is a list of override tags, each one related to a specific component. Several override tags can relate to the same component, but each override tag can only relate to one component.

If you do not specify the :OVERLSTP. tag, the names following the :OVERLST. tag are used for prompting. You do not have to list the override names twice. You will be prompted only once for any name listed on both tags.

The :OVERLST. specification is required in the product area of both the base product parameter file and the separate override file. It is not allowed in the component override section of either file.

:OVERLST. must always be placed after the :OVERLSTP. tag in the base PPF.

Component Area

:compname.
is an uppercase delimiter for the part of the PPF describing the component named. *compname* is one of the components named in :COMPLST. The end of the component part of the PPF for this component is delimited by a corresponding :END. tag.

Control Options

:CNTRLOP.
is the delimiter for the component section of the PPF defining general parameters for the component.

:CNTRL. *control_file_name*
is the filename of the control file to be used to service, assemble, and/or build this component of the product.

:SLVI. *p1/ss1 p2/ss2 ... pn/ssn*

:SLVI
is a list of the system level and version indicators used as a prefix and suffix, concatenated with the numeric portion of the APAR number, to form the filetype of the update file.

p/ss
is a word that consists of two values separated by a slash. The first value, represented by a p, is the single character prefix in the update file filetype. The second value, represented by ss, is the 2-character suffix in the update file filetype.

:NLS. *langid*
is specified to be one of the following: AMENG, KANJI, UCENG, PORTG, FRANC, or GER. For CP and CMS the default is AMENG.

:LTO. YES|NO

:LTO.
The Last Text Only option erases lower level text decks during receive. Duplicate text decks are never created, regardless of the setting of the LTO option (yes or no).

YES
indicates that during VMFREC a search is made for other text decks with the same filename and filetype prefix in the target minidisk string. If other decks exist, the PTF information in each text deck prologue is checked and only the latest text deck, including the one just read in, is kept on the DELTA disk. The other levels are replaced with text deck shells. The latest text deck includes the accumulation of all service in the decks being replaced. If this is not the case an error is included in the receive exception log. This requires write access to all DELTA disks on which text decks may exist.

NO
indicates that existing text decks for different PTFs are left as they are. If a text deck exists for the same PTF the new one is not kept.

:CKTXT. YES|NO

:CKTXT.
Check for full text at apply. The value of this tag may be overridden by using the CHECK or NOCHECK option on VMFAPPLY. CHECK is equivalent to :CKTXT. YES. NOCHECK is equivalent to :CKTXT. NO.

YES
indicates that when VMFAPPLY is complete, if a PTF which has been applied is a text shell and not a full text deck including executable code, an error should be placed in the Apply Exception Log.

NO
indicates that the application of a PTF which is a text shell will not cause an error in the exception log.

:CKUPD. YES|NO

:CKUPD.
Check and apply PTFs whether or not they have already been applied. A filemode of 5 on the $PTFPART list indicates that a PTF has been applied.

YES
indicates that a filemode of 5 on a $PTFPART list is not sufficient to prevent a PTF from being applied. Additional processing is performed for each part in the list. This makes it possible for the same PTF to be processed more than once.

NO
indicates that filemode 5 on the $PTFPART list will prevent the same PTF from being processed more than once. If the PTF has already been successfully applied as indicated by its filemode all further processing for this PTF is bypassed.

:CKAUX. YES|NO

:CKAUX.
Check composite AUX at build. The value of this tag may be overriden by using the CHECK or NOCHECK option on VMFBLD. CHECK is equivalent to :CKAUX. YES and NOCHECK is equivalent to :CKAUX. NO.

YES
indicates that all AUX and update entries in the text deck should be verified against the AUX control structure to be sure the correct deck has been located.

NO
indicates that only the entries in the AUX file used to locate the text deck will be verified against the text deck prolog. Additional entries in the text deck prolog are not verified for faster completion of the build process.

:CKREQ. YES|NO

:CKREQ.
Check requisites at build. The value of this tag may be overridden by using the CHECK or NOCHECK option on VMFBLD. CHECK is equivalent to :CKREQ. YES and NOCHECK is equivalent to :CKREQ. NO.

YES
indicates that requisite information chains will be verified to assure that all required PTFs related to the ones applied are included in the system being built. The update files contain entries for PREREQ, CO-REQ and IF-REQ.

PREREQs and CO-REQs describe dependencies in the same product. If the dependency is in the same component, the 7-character APAR number is listed (for example, VM12345). If the dependency is in another component of the same product the APAR number and component ID are included in parentheses following the PTF number, for example, UV54321(VM12345-56630130).

If the dependency is in another product, an IF-REQ is used. An IF-REQ contains a PTF number rather than an APAR number. The APAR number and component ID are included in parentheses following the PTF number, for example, UV54321(VM12345-56630130).

NO
indicates that requisite information chains as contained in the text will not be verified during the VMFBLD EXEC.

:UPDTID. *updateid*
identifies the filetype of an AUX file and should match a filetype listed in the control file. This is the level at which AUX files will be updated or created as PTFs are applied. This *updateid* will be used unless you override it by providing an *updateid* as an argument to VMFAPPLY.

:USEREXIT. *exitname*

:USEREXIT.
identifies the user EXEC for setup and cleanup to be called at beginning and end of each service function (VMFREC, VMFAPPLY, and so on).

exitname
is the name of the user EXEC to be called. The first parameter is the uppercase name of the invoking EXEC. The second parameter is either SET-UP or CLEAN-UP, indicating initialization or termination. At termination the third parameter is the return code as determined by the invoking EXEC. The return code can be changed by the exit. Functions such as link and detach can be invoked by the exit. VMFUECP, VMFUECMS, and VMFUEDV are the default exits provided by IBM. These exits are no-ops.

:MERGE. *symbol1 symbol2 ... symboln*

:MERGE.
is a list of symbolic disk strings (for example DELTA, APPLY) to be merged by VMFREC when the product or component being received already exists on the receive target disk. If the target disk does not contain the product or component being received, the merge does not take place.

VMFREC accomplishes the merge by moving the contents of the disk before the last in the string to the last disk in the string, then erasing the disk before the last. It repeats this process with the next pair of disks, moving one disk to the left in the string each time, until the first disk in the string is clear.

Note: If you want to prevent merging from taking place, see Appendix G, "Controlling Disk String Merges" on page 861.

symbol
is the symbolic name of a minidisk string such as DELTA1, APPLY, or LOCAL1. If the product or component is present, as indicated in a file with the filetype $MR*xxxxx*, on the first disk in any disk string in the list, all disk strings in the list are merged.

Minidisk Assignments:

Usage Notes:

- Considerations for read-only disks in the product parameter file:

  Disks appearing in the MDA section of the $PPF may be specified as read-only by coding a slash (/) immediately following the disk address in the symbolic disk string. For example:

```
DELTA 594 291 294/
```

  indicates that the 294 disk is to be read-only and it will be accessed as mode/mode by VMFSETUP.

- The first specification wins:

  If the disk address appears in a given section (receive, apply, and so on) more than once, its first specification determines its access status.

- Disks currently accessed R/W:

  If a disk is currently accessed R/W it is released and then accessed mode/mode. Upon performing a VMFSETUP restore it will once again be R/W.

  If the disk is R/W and empty (no CMS files on it) the mode/mode fails and a message is issued when the using application program attempts to access it.

- Merge considerations:

  Do not use read-only disks in symbolic disk strings identified as strings to be merged. Merging requires write access.

:MDA.
is an uppercase delimiter for the section describing the minidisk assignments.

TASK *dddd1 dddd2 ... ddddn*

TASK
lists the service EXECs build disk. The TASK disk is accessed after the A-disk and before the component data base disks. For more information on using the TASK statement to set up a separate disk for the service EXECs, see the "Usage Notes" section of the VMFREC EXEC description (on page /USAGE/). The symbolic disk name TASK must start in column 1.

dddd
is the 4-digit disk address of a system minidisk.

SYSTEM *dddd1 dddd2 ... ddddn*

SYSTEM
lists the disks required for your working system environment. The symbolic disk name SYSTEM must start in column 1.

dddd
is the 4-digit disk address of a system minidisk.

BASE*n dddd*

BASE1 ... BASE*n*
are the BASE disks created at installation time. You may define additional BASE disks if you have specific needs. The symbolic disk name BASE must start in column 1.

dddd
is the 4-digit disk address of a BASE minidisk.

DELTA*n dddd1 dddd2 ... ddddn*

DELTA1 ... DELTA*n*
are disks containing service from the PUT. You may define additional DELTA disks. The symbolic disk name DELTA must start in column 1.

dddd
is the 4-digit disk address of a DELTA minidisk.

APPLY*n dddd1 dddd2 ... ddddn*

APPLY1 ... APPLY*n*
identifies the control structure to be built. AUX files and update shells are created on the APPLY minidisks. There is only one symbolic string of APPLY minidisks. The symbolic disk name APPLY must start in column 1.

dddd
is the 4-digit disk address of an APPLY minidisk.

LOCAL*n dddd1 dddd2 ... ddddn*

LOCAL1 ... LOCAL*n*
are disks for local service and COR service. You may define additional LOCAL disks The symbolic disk name LOCAL must start in column 1.

dddd
is the 4-digit disk address of a LOCAL minidisk.

BUILD*n dddd*

BUILD1 ... BUILD*n*
are the new system residence disks for CP nucleus build. You may define additional BUILD disks. The symbolic disk name BUILD must start in column 1.

Warning: The :BUILD1. tag does not force the nucleus to be built on the specified disk. Make sure that the HCPSYS ASSEMBLE file (for CP), the DMSNGP profile (for CMS), or the saved system information panel (GRP121, for GCS), indicates the disk where you want to build the nucleus.

dddd
is the 4-digit disk address of a BUILD minidisk.

EXECs

:PRTFNC.
is an uppercase delimiter for the section describing the list of the types of parts serviced with the functional EXEC assignments and symbolic target disk assignments.

:RECSER.
is an uppercase delimiter for the list of functional assignments related to the Receive function (VMFREC EXEC). This tag is followed by a list which contains one entry for each possible tape file for the component on the PUT or COR service tape.

Each tape file contains all the parts of the same or a related type. The attribute that binds the types of part in a particular tape file is that they all require the same Receive processing and target minidisk symbol. Each part is in a separate CMS file within the tape file.

tapfil execname target

tapfil
is an uppercase mnemonic that symbolizes the content of the tape file it represents. This is the first symbol in each entry for each tape file on the PUT or COR service tape. This symbol may start in any column.

execname
is the name of the EXEC that processes the types of parts found in the tape file represented by this entry. IBM provides four part handlers for the Receive function. VMFRCUPD processes update files. VMFRCTXT processes text decks. VMFRCAXL processes apply and exclude lists. VMFRCCOM processes all other parts by simply putting them to the target disk.

An EXEC name starting with an asterisk causes the associated tape file not to be received.

target
is an uppercase symbol corresponding to the minidisk assignment section of the PPF that identifies where the parts are to be placed as they are received from the PUT.

:APP.
is an uppercase delimiter for the list of functional assignments related to the apply function (VMFAPPLY EXEC). This tag is followed by a list which contains one entry for each type of part for the component.

partype execname rename

partype
is an uppercase part filetype or part filetype prefix when terminated by an asterisk. If the partype symbol is TXT*, all parts with a filetype starting with the characters TXT are processed by the part handler name in this entry. If the symbol is MACRO, all parts with the filetype MACRO are processed by the part handler name in this entry. This symbol may start in any column.

execname
is the name of the EXEC that processes the types of parts found with the filetype represented in this entry. Parts with a filetype for which no partype symbol appears in the list will be processed by the common part type handler VMFAPCOM. Two other part handlers are provided by IBM for the Apply function. VMFAPTXT processes text decks, VMFAPNLS language versions of text decks.

rename
is the new filetype to be given to those parts found to have an existing filetype matching the partype symbol in the entry, but not supported by VMFBLD. Parts supported by VMFBLD are not renamed. Whether parts are supported is determined by finding the part in the build list. Parts in the build list are located by PTF number. Parts not found in the build list are copied with a new filetype of *rename*.

:BLD.
is an uppercase delimiter for the list of functional assignments related to the build function (VMFBLD EXEC). This tag is followed by a list which contains one entry for each type of part which is built from a collection of other parts in the component. This symbol may start in any column.

bldlst execname target

bldlst
is an uppercase filename for the list (build list) of parts to be included in the part being built. For a nucleus this is a load list.

execname
is the filename of the EXEC that processes the build list.

target
is the symbolic name corresponding to the minidisk assignment section of the PPF that identifies where the part being built is to be placed. The minidisk address assigned to the symbol specified here should match the SYSRES address specified on the SYSRES macro in the HCPSYS file.

:END.
identifies the end of the current component part. The next component in the PPF starts with the next *:compname.* from the :COMPLST. tag.

Override Area

:overname. compname ppfname **[NEWNAME | OLDNAME]**

:overname.
identifies the beginning of the override section for a component. This statement does not appear in the temporary PPF.

compname
is the name of the component specified in the base PPF :COMPLST. that is to be overridden.

ppfname
is the name of the PPF that is to be overridden. If *ppfname* is not specified, *overname* and *compname* must be in the same file (that is, the component override is part of the base PPF).

NEWNAME
indicates that the filename of the temporary PPF should be the filename of the override file. This option cannot be used in the override section of the base PPF (that is, when override and base PPF are in the same file). This option cannot be used during VMFREC.

OLDNAME
indicates that the filename of the temporary PPF should be the original filename of the base PPF file. OLDNAME is the default if neither OLDNAME or NEWNAME is specified.

:MDA. [REPLACE | UPDATE]
:RECSER. [REPLACE | UPDATE]
:APP. [REPLACE | UPDATE]
:BLD. [REPLACE | UPDATE]

:MDA.
introduces the overrides for the minidisk assignment section of the component area.

:RECSER.
introduces the overrides for the list of functional assignments related to the receive function (VMFREC EXEC).

:APP.
introduces the overrides for the list of functional assignments related to the Apply function (VMFAPPLY EXEC).

:BLD.
introduces the overrides for the list of functional assignments related to the build function (VMFBLD EXEC).

UPDATE
indicates that the entries following the tag in the override section will overlay the corresponding entry in the base PPF. Any overlaid entry appears as a comment in the temporary PPF. If the entry does not exist in the base, the entry will be added at the end of the section in the base PPF. UPDATE is the default if neither UPDATE or REPLACE is specified.

REPLACE
indicates that all entries in the base PPF are to be removed and replaced by the entries in the override PPF. The removed entries do not appear as comments in the temporary PPF.

*
Comments or a group of comments beginning with a single asterisk are merged into the temporary PPF immediately preceding the next non-comment statement.

**
Comments or a group of comments beginning with a double asterisk are merged into the temporary PPF immediately following the previous non-comment statement.

***
Comments beginning with three asterisks do not appear in the temporary override file.

**** ...
Comments beginning with more than three asterisks are handled as if they began with one asterisk.

The Product Parameter Override File

The product parameter override file is a separate file containing tags that you want to override the tags in the base product parameter file when you build or service your system. The override file permits you to create a new version of the product without creating a complete new product parameter file.

The only difference between a separate override file and the override section of a base PPF is that the separate override file begins with an optional :OVERLSTP. tag and an :OVERLST. tag. In a base PPF, these tags come immediately after the :COMPLST. tag at the beginning of the file, not in the override section.

The filename of the override file is the name of the new version of the product. The filetype is $PPF.

The following example shows the override file used with the base file 56643089 $PPF to create an alternate product identifier (or "alias") for VM/XA SP. Since the alternate product identifier is VMXA, the override file is called VMXA $PPF.

Note: If you want to invoke any overrides from the override section of the base product parameter file, for example, the COR service overrides, you must copy them to the separate override file.

Save override files on the CMS alternate LOCAL1 (395) disk. IBM suggests that you should also save the updated product parameter file there.

Figure 26 is an example of a product parameter override file.

```
* This is the alias override for the VM/XA Product 56643089
:OVERLST. CP CMS DV GCS
*** The overlst statement will not appear on the temporary composite
*** product parameter file.
:CP.  CP 56643089          * Get the base unmodified PPF from the CP
***                          * file named 56643089 $PPF
:END.                        * End of current component section
:CMS. CMS 56643089         * Get the base unmodified PPF from the CMS
***                          * file named 56643089 $PPF
:END.                        * End of current component section
:DV. DV 56643089           * Get the base unmodified PPF from the DVF
***                          * file named 56643089 $PPF
:END.                        * End of current component section
:GCS. GCS 56643089         * Get the base unmodified PPF from the CMS
***                          * file named 56643089 $PPF
:END.                        * End of current component section
```

Figure 26. Example of a Product Parameter Override File (VMXA $PPF)

The Temporary Product Parameter File

VMFREC, VMFAPPLY, VMFHASM, VMFASM, VMFSETUP, VMFNLS, and VMFBLD all invoke the VMFOVER EXEC to create the temporary product parameter file from the product parameter file and sometimes the product parameter override file. The temporary product parameter file is a copy of the product area and one component area of the base product parameter file, with the tags from the override section or override file substituted where appropriate. If you specify the UPDATE option (the default) in the overrides, the overridden tags are included in the temporary PPF as comments. If you specify the REPLACE option, the overridden tags are not included. All the installation and service EXECs use this copy instead of the base PPF.

If you use a separate override file and specify the NEWNAME option on the *:compname.* tag in the override file the filename of the copy is the filename of the override file. Otherwise, it is the filename of the base PPF. The filetype of the copy is $PPFTEMP.

To see the temporary product parameter file before you run a service EXEC, invoke VMFOVER from the command line. You can also examine the temporary product parameter file after running the service EXECs. It is not erased.

Many different temporary product parameter files can be created from the base file on page 384, 56643089 $PPF. In the simplest example, if you issue this command:

```
vmfover 56643089 cp ■
```

you will get a temporary file just like the CP section of the base file, like the file in Figure 27 on page 404.

```
*=============================================
*====>   * Temporary override of CP component
*====>   * in PPF of product 56643089
*====>   * by CP override section
*====>   * of the PPF for the product 56643089
*=============================================
*********************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1989
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083
*********************************************************************
*=====================================================================
*            Product Parameter File for VM/XA SP Release 2.1
*=====================================================================
*
* The following APAR's have been applied to this file:
*
*  APAR      Date       Description
*  =======   ========   ===========================================
*
*
*=====================================================================
*            NOTE: All tags must be in uppercase.
*=====================================================================
```

Figure 27 (Part 1 of 4). 56643089 $PPFTEMP (without Overrides)

```
*====================================================================
* Start of Product Header - List of Components in VM/XA SP Release 2.1
*====================================================================
:COMPLST.      CP    CMS    DV    GCS
:OVERLSTP.
:OVERLST.
*====================================================================
* End of Product Header
*====================================================================
*====================================================================
:CP.                          * Start of Parameters for CP
*====================================================================
:CNTRLOP.                     * Section one - Control Options
**                            * DO NOT DELETE THESE CONTROL TAGS
* TAG       VALUE(S)          * DESCRIPTION
**
:CNTRL.     HCPXA             * Control Filename
:UPDTID.    AUXXA             * Update level identifier for building AUX
:SLVI.      A/HP A/PP         * System level and version indicators
:NLS.       AMENG             * System Language
:USEREXIT.  VMFUECP           * User exit EXEC.  Called before initial
**                            * accesses and after final access retore.
**                            * 2 paramters passed will indicate calling
**                            * EXEC and SET-UP or CLEAN-UP.
:MERGE.     DELTA1 APPLY      * Disk strings to be merged at Receive
:LTO.       NO                * Erase lower level TEXT decks at Receive
:CKTXT.     YES               * Check for full text at Apply
:CKUPD.     NO                * Check update fm regardless of $PTFPART fm
:CKAUX.     YES               * Check composite AUX at Build
:CKREQ.     YES               * Check Requisites at Build
*====================================================================
:MDA.                         * Section two - Mini-Disk Assignments
**                            * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS          * DESCRIPTION
**
TASK        5E5               * Build disks accessed before rest of dbase
LOCAL1      295               * Disks for local service
*LOCALn                       * You may define additional LOCAL disks
APPLY       492 291 292       * Control structure built by service execs
*APPLYn                       * You may define additional APPLY disks
DELTA1      594 591 294       * CP service from TAPE
*DELTAn                       * You may define additional DELTA disks
BASE1       194               * CP text,loadlist,control file,maclibs
BASE2       394               * CP source,macro defs,and copy files
*BASEn                        * You may define additional BASE disks
*BUILD1     423               * New SYSRES for CP Nucleus Build
BUILD2      49D               * New CMS system disk for AMENG HELP files
BUILD3      49C               * New CMS system disk for UCENG HELP files
BUILD4      490               * New CMS system disk for nucleus etc.
```

Figure 27 (Part 2 of 4). 56643089 $PPFTEMP (without Overrides)

```
*BUILDn                        * You may define additional BUILD disks
SYSTEM     191                 * Disks reqd for your system environment
*===============================================================================
:PRTFNC.                       * Section three - Part type/Function List
*===============================================================================
:RECSER.                       * Receive (Service)
**
* TAPEFILE EXEC     TARGET    * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1    * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1    * PTF Parts Lists
  UPDT     VMFRCUPD DELTA1    * Update files for CP ASSEMBLE source
  TEXT     VMFRCTXT DELTA1    * Text replacement modules for CP
**                            * Text shells created on target disk
**                            * Update shells created on APPLY disk
  MACAUX   VMFRCAUX DELTA1    * AUX files for updates to non-ASSEM parts
  MACUPDT  VMFRCUPD DELTA1    * Update files for non-ASSEMBLE parts
  SOURCE   VMFRCCOM DELTA1    * New Source (assemble,macro,copy,$exec...)
  MACLIB   VMFRCCOM DELTA1    * Maclibs are not replaced on the base
  EXEC     VMFRCCOM DELTA1    * Replacement parts and regenerated
**                            * parts other than maclibs, text, modules
  HELP     VMFRCCOM BUILD2    * AMENG HELP files go directly to the
**                            * AMENG help Build disk
  HELP     VMFRCUPP BUILD3    * UCENG HELP files go directly to the
**                            * UCENG help Build disk
  RESV    *VMFRCCOM DELTA1    * Reserved parts
  CPMOD    VMFRCCOM BUILD4    * CP modules targeted for system disk
  PTFS     VMFRCCOM DELTA1    * PTF numbered parts other than TXT
*===============================================================================
:APP.                          * Apply Function
**
* PARTTYPE EXEC     RENAME    * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT      * Create AUX and rename appropriate update
**                            * files to fm5 for names in loadlist.
**                            * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG  * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI  * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG  * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG  * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC  * For French NLS support
  TGR*     VMFAPNLS TXTGER    * For German NLS support
  AUXXA    VMFAPAUX           * For MACRO and EXEC AUX files
  AXA*     VMFAPAUX AUXXA     * For MACRO and EXEC AUX files
  EXEC     VMFAPCOM           *
  MACRO    VMFAPCOM           *
** .       VMFAPCOM           * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM           * If a part type is not listed
** .       VMFAPCOM           * VMFAPCOM is called by default
*===============================================================================
```

Figure 27 (Part 3 of 4). 56643089 $PPFTEMP (without Overrides)

```
:BLD.                          * Build Function
**
* BUILDLST EXEC     TARGET    * DESCRIPTION
**
  CPLOAD   VMFBDNUC *BUILD1   * Create CP nucleus based on loadlist
**                             * Target disk symbol agrees with
**                             * disk address in HCPSYS ASSEMBLE
**                             * on the SYSRES macro. Defaults to
**                             * what is in HCPSYS ASSEMBLE if no
**                             * target disk is specified.
**
  CPYSYSCP VMFBDCPY BUILD4     * Copy TEXT decks that do not become
**                             * part of a MODULE to the system disk
**
  CPYBTHCP VMFBDCPY DELTA1     * Copy TEXT decks used by UTILITY and
**                             * listed in the load list, CPLOAD EXEC,
**                             * to the DELTA disk. VMFAPPLY only
**                             * copies the ones not listed in the
**                             * load list.
**
* CPYUTLCP VMFBDCPY DELTA1     * Copy TEXT decks used by UTILITY which
**                             * are not listed in the load list,
**                             * CPLOAD EXEC. This list would be used
**                             * if you have local modifications and
**                             * you choose not to run VMFAPPLY. All
**                             * TEXT decks for which you have received
**                             * service and/or have local mods would
**                             * be copied as a result. You need to remove
**                             * the asterisk before CPYUTLCP before
**                             * running VMFBLD.
**                             * Use UTILITY to build CP Utilities
*================================================================================
:END.                          * End of Parameters for CP
```

Figure 27 (Part 4 of 4). 56643089 $PPFTEMP (without Overrides)

If you issue

vmfover 56643089 corcp ■

you will get a temporary file with overrides from the override section of the base file, like the file in Figure 28 on page 408.

Note: When you issue VMFREC or VMFAPPLY, use a component name of CP and the COR option instead of the CORCP component name to create this temporary file.

```
*=============================================
*====>   * Temporary override of CP component
*====>   * in PPF of product 56643089
*====>   * by CORCP override section
*====>   * of the PPF for the product 56643089
*=============================================
*************************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1989
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083
*************************************************************************
*=======================================================================
*            Product Parameter File for VM/XA SP Release 2.1
*=======================================================================
*
* The following APAR's have been applied to this file:
*
*  APAR      Date       Description
*  =======   ========   ===========================================
*
*
*=======================================================================
*            NOTE: All tags must be in uppercase.
*=======================================================================
*=======================================================================
* Start of Product Header - List of Components in VM/XA SP Release 2.1
*=======================================================================
:COMPLST.      CP    CMS    DV    GCS
:OVERLSTP.
:OVERLST.
*=======================================================================
* End of Product Header
*=======================================================================

*=======================================================================
:CP.                        * Start of Parameters for CP
*=======================================================================
:CNTRLOP.                   * Section one - Control Options
**                          * DO NOT DELETE THESE CONTROL TAGS
* TAG      VALUE(S)         * DESCRIPTION
**
:CNTRL.    HCPXA            * Control Filename
:UPDTID.   AUXXA            * Update level identifier for building AUX
:SLVI.     A/HP A/PP        * System level and version indicators
:NLS.      AMENG            * System Language
:USEREXIT. VMFUECP          * User exit EXEC.  Called before initial
**                          * accesses and after final access retore.
**                          * 2 paramters passed will indicate calling
**                          * EXEC and SET-UP or CLEAN-UP.
*=======================================================================
```

Figure 28 (Part 1 of 4). 56643089 $PPFTEMP (with Corrective Service Overrides)

```
:MERGE.    APPLYX DELTAX     * Merge dummy APPLY and DELTA strings
*:MERGE.    DELTA1 APPLY      * Disk strings to be merged at Receive
:LTO.      NO                * Erase lower level TEXT decks at Receive
:CKTXT.    YES               * Check for full text at Apply
:CKUPD.    NO                * Check update fm regardless of $PTFPART fm
:CKAUX.    YES               * Check composite AUX at Build
:CKREQ.    YES               * Check Requisites at Build
*=================================================================
:MDA.                        * Section two - Mini-Disk Assignments
**                           * STRING NAMES MUST START IN COLUMN 1
* STRNGNAM MINIDISKS         * DESCRIPTION
**
TASK       5E5               * Build disks accessed before rest of dbase
LOCAL1     295               * Disks for local service
*LOCALn                      * You may define additional LOCAL disks
APPLY      492 291 292       * Control structure built by service execs
*APPLYn                      * You may define additional APPLY disks
DELTA1     594 591 294       * CP service from TAPE
*DELTAn                      * You may define additional DELTA disks
BASE1      194               * CP text,loadlist,control file,maclibs
BASE2      394               * CP source,macro defs,and copy files
*BASEn                       * You may define additional BASE disks
*BUILD1    423               * New SYSRES for CP Nucleus Build
BUILD2     49D               * New CMS system disk for AMENG HELP files
BUILD3     49C               * New CMS system disk for UCENG HELP files
BUILD4     490               * New CMS system disk for nucleus etc.
*BUILDn                      * You may define additional BUILD disks
SYSTEM     191               * Disks reqd for your system environment
*=================================================================
APPLYX     492 291           * Define dummy APPLY string
DELTAX     594 591           * Define dummy DELTA string
**                           * During corrective service merges are per-
**                           * formed based on these dummy strings.  The
**                           * intermediate disks act as collectors for
**                           * corrective service until the next PUT.
:PRTFNC.                     * Section three - Part type/Function List
*=================================================================
:RECSER.                     * Receive (Service)
**
* TAPEFILE EXEC     TARGET   * DESCRIPTION
**
  AXLIST   VMFRCAXL DELTA1   * Apply and Exclude lists
  PARTLST  VMFRCCOM DELTA1   * PTF Parts Lists
  UPDT     VMFRCUPD DELTA1   * Update files for CP ASSEMBLE source
  TEXT     VMFRCTXT DELTA1   * Text replacement modules for CP
**                           * Text shells created on target disk
**                           * Update shells created on APPLY disk
  MACAUX   VMFRCAUX DELTA1   * AUX files for updates to non-ASSEM parts
  MACUPDT  VMFRCUPD DELTA1   * Update files for non-ASSEMBLE parts
  SOURCE   VMFRCCOM DELTA1   * New Source (assemble,macro,copy,$exec...)
  MACLIB   VMFRCCOM DELTA1   * Maclibs are not replaced on the base
  EXEC     VMFRCCOM DELTA1   * Replacement parts and regenerated
**                           * parts other than maclibs, text, modules
```

Figure 28 (Part 2 of 4). 56643089 $PPFTEMP (with Corrective Service Overrides)

```
  HELP     VMFRCCOM BUILD2   * AMENG HELP files go directly to the
**                           * AMENG help Build disk
  HELP     VMFRCUPP BUILD3   * UCENG HELP files go directly to the
**                           * UCENG help Build disk
  RESV    *VMFRCCOM DELTA1   * Reserved parts
  CPMOD    VMFRCCOM BUILD4   * CP modules targeted for system disk
  PTFS     VMFRCCOM DELTA1   * PTF numbered parts other than TXT
*=============================================================================
:APP.                        * Apply Function
**
* PARTTYPE EXEC     RENAME   * DESCRIPTION
**
  TXT*     VMFAPTXT TEXT     * Create AUX and rename appropriate update
**                           * files to fm5 for names in loadlist.
**                           * Other TXT parts renamed to TEXT
  TAM*     VMFAPNLS TXTAMENG * For American English NLS support
  TKA*     VMFAPNLS TXTKANJI * For KANJI NLS support
  TUC*     VMFAPNLS TXTUCENG * For Uppercase English NLS support
  TPO*     VMFAPNLS TXTPORTG * For Brazilian Portuguese NLS support
  TFR*     VMFAPNLS TXTFRANC * For French NLS support
  TGR*     VMFAPNLS TXTGER   * For German NLS support
  AUXXA    VMFAPAUX          * For MACRO and EXEC AUX files
  AXA*     VMFAPAUX AUXXA    * For MACRO and EXEC AUX files
  EXEC     VMFAPCOM          *
  MACRO    VMFAPCOM          *
** .       VMFAPCOM          * ALL PARTS ARE LISTED BY FILETYPE
** .       VMFAPCOM          * If a part type is not listed
** .       VMFAPCOM          * VMFAPCOM is called by default
*=============================================================================
:BLD.                        * Build Function
**
* BUILDLST EXEC     TARGET   * DESCRIPTION
**
  CPLOAD   VMFBDNUC *BUILD1  * Create CP nucleus based on loadlist
**                           * Target disk symbol agrees with
**                           * disk address in HCPSYS ASSEMBLE
**                           * on the SYSRES macro. Defaults to
**                           * what is in HCPSYS ASSEMBLE if no
**                           * target disk is specified.
**
  CPYSYSCP VMFBDCPY BUILD4   * Copy TEXT decks that do not become
**                           * part of a MODULE to the system disk
**
  CPYBTHCP VMFBDCPY DELTA1   * Copy TEXT decks used by UTILITY and
**                           * listed in the load list, CPLOAD EXEC,
**                           * to the DELTA disk. VMFAPPLY only
**                           * copies the ones not listed in the
**                           * load list.
**
```

Figure 28 (Part 3 of 4). 56643089 $PPFTEMP (with Corrective Service Overrides)

```
* CPYUTLCP VMFBDCPY DELTA1   * Copy TEXT decks used by UTILITY which
**                          * are not listed in the load list,
**                          * CPLOAD EXEC. This list would be used
**                          * if you have local modifications and
**                          * you choose not to run VMFAPPLY. All
**                          * TEXT decks for which you have received
**                          * service and/or have local mods would
**                          * be copied as a result. You need to remove
**                          * the asterisk before CPYUTLCP before
**                          * running VMFBLD.
**                          * Use UTILITY to build CP Utilities
*============================================================================
*============================================================================
:END.                       * End of Parameters for CP
```

Figure 28 (Part 4 of 4). 56643089 $PPFTEMP (with Corrective Service Overrides)

If you have a VM/XA $PPF file like the one in Figure 26 on page 403 and you issue:

vmfover vmxa cp ■

you will get a temporary file, that looks just like the file in Figure 27 on page 404.

The Program Update Tape (PUT)

The format of each volume of the PUT is as follows:

- **The first tape file** contains the PUT DOCUMENT, the VMSERV EXEC, and the latest tested level of the service EXECs (with PTF-numbered filenames).
- **The second tape file** contains a Memo to Users for each program product service on the volume, and the latest tested level of the product parameter file for each product on the tape.
- **The remaining tape files** contain service files for the program products on that volume.

Figure 29 on page 412 shows the format of the PUT tape.

```
------------------------------------------------------------------
        --- Program Update Tape (PUT) Format for VM/XA SP 2.1 ---
------------------------------------------------------------------


------------------------------------------------------------------
|                        --- Tape File 1 ---                      |
------------------------------------------------------------------
PUT      8801             <=== PUT Descriptor File
PUT      Document         <=== PUT Content and Instructions
VMSERV   EXEC             <=== Compatible Service EXEC
:
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|                        --- Tape File 2 ---                      |
------------------------------------------------------------------
prodid1  crllnn1          <=== Program Level File
prodid1  MEMO             <=== Instructions for product
prodid1   $PFnnnnn        <=== Product Parameter File
prodid2  crllnn0
prodid2  MEMO
prodid3  crllnn1
prodid3  MEMO
prodid3   $PFnnnnn        <=== Product Parameter File
prodid4  crllnn1
prodid4  MEMO
prodid4   $PFnnnnn        <=== Product Parameter File
:
    ========================== Tape Mark ==========================


------------------------------------------------------------------
|                 --- Product 1 File 1 - Header ---               |
------------------------------------------------------------------
prodid1  crllnn1
prodid1  $PUTnnnn1        <=== Product Contents Directory
                          <=== (Flag byte 1 indicates new format)
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 2 -Component 1 - Part 1 ---              |
------------------------------------------------------------------
COMP1 PART1A              <=== Group processed according to Part 1
COMP1 PART1B                   as described in PPF in terms of process
COMP1 PART1C                   destination.
COMP1 PART1D
```

Figure 29 (Part 1 of 3). Format of the Program Update Tape

```
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 3 -Component 1 - Part 2 ---              |
------------------------------------------------------------------
COMP1 PART2E              <=== Processed according to Part 2 in PPF
COMP1 PART2F
COMP1 PART2G
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 4 -Component 1 - Part 3 ---              |
------------------------------------------------------------------
COMP1 PART3H              <=== Processed according to Part 3 in PPF
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 5 -Component 2 - Part 1 ---              |
------------------------------------------------------------------
COMP2 PART1A              <=== Processed according to Part 4 in PPF
COMP2 PART1B
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 6 -Component 2 - Part 2 ---              |
------------------------------------------------------------------
COMP2 PART2C              <=== Processed according to Part 5 in PPF
COMP2 PART2D
COMP2 PART2E
COMP2 PART2F
COMP2 PART2G
    ========================== Tape Mark ==========================
------------------------------------------------------------------
|                --- Product 2 - File 1 - Header ---              |
------------------------------------------------------------------
prodid2  crllnn0          <=== Flag byte 0 indicates compatible
prodid2  EXEC                  format
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 2 -Component 1 - Part 1 ---              |
------------------------------------------------------------------
COMP1 PART1A              <=== Group processed by product exec
COMP1 PART1B
    ========================== Tape Mark ==========================

------------------------------------------------------------------
|               --- File 3 -Component 1 - Part 2 ---              |
------------------------------------------------------------------
COMP1 PART1C              <=== Group processed by product exec
COMP1 PART1D
COMP1 PART1E
COMP1 PART1F
```

Figure 29 (Part 2 of 3). Format of the Program Update Tape

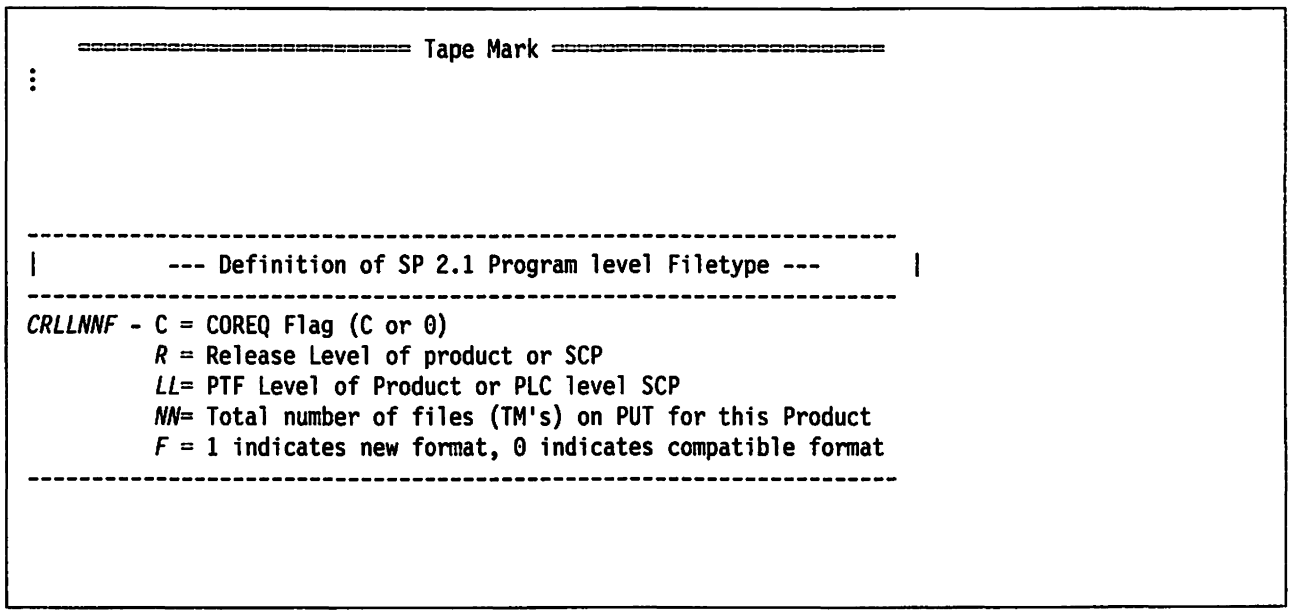


Figure 29 (Part 3 of 3). Format of the Program Update Tape

Although the VMSERV EXEC and service EXECs for each product are still provided on the PUT tape, you must use the new service EXECs, VMFREC, VMFAPPLY, and VMFBLD, to receive and apply service to CP, CMS, GCS, and the dump viewing facility. These EXECs are loaded from the VM/XA SP product tape when you install the system. You can still use VMSERV to receive and apply service to other program products. See Chapter 10 through Chapter 18 for examples of using the new EXECs.

IBM packages service for VM/XA SP as service level updates. Periodically, or whenever you request it, IBM ships a service level update. This includes cumulative preventive maintenance since the last release of VM/XA SP.

You may find that there is more than one preventive update on a PUT tape.

The Corrective Service Tape (COR)

For new format products, the format of the COR tape is exactly the same as the format of the PUT tape, except that only the tape files containing parts that actually need corrective service are included, and the VMSERV EXEC is not included.

Chapter 9. Preparing For Service

This chapter describes how to establish an alternate level of build disks to use when servicing your system.

Step 1. Minidisk Accesses

Before you begin processing service, you must have access to the service tools on the TASK disk (5E5). You should include this access in your PROFILE EXEC:

```
access 5E5 b
```

Step 2. Establish an Alternate Set of Build Disks

Before you service your VM/XA System Product Release 2.1 system, you should create an alternate level of build disks. This isolates the changed system so it can be tested before you make it available to your general user community.

1. Use the DASD DUMP Restore program to copy the current BUILD1 disk to its corresponding alternate disk:

   ddr ■
   VM/XA SYSTEM PRODUCT DASD DUMP RESTORE PROGRAM
   ENTER:

   sysprint cons ■
   ENTER:

   This command tells DDR to send program messages to your console.

   input 190 *devtype* **MNT190** ■
   ENTER:

   190 is your current CMS system disk. *devtype* is the device type of the DASD volume where 190 is located.

   output 490 *devtype* **MNT490** ■
   ENTER:

   490 is your alternate CMS system disk. *devtype* is the device type of the DASD volume where 490 is located.

   copy 000 *endcyl* ■

   The value for *endcyl* depends on the device type of your 190 disk:

Device Type	*endcyl*
3350	73
3375	112
3380	71
3390	67

```
DMKDDR711R Volid Read is MNT190.  Do you
                wish to continue?
yes ■
ENTER NEXT EXTENT OR NULL LINE:
■
⋮
END OF COPY
ENTER:
■
END OF JOB
```

You have not yet changed the label of the 490 minidisk. (You will do so in substep 2.) This prompt will not always appear.

2. Relabel the 490 minidisk:

```
access 490 t ■
Ready; T=n.nn/n.nn hh:mm:ss
format 490 t (label ■
ENTER LABEL:
mnt490 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Access 490 *after* the system disk. You *must* use the **label** option. If you don't, you will erase all the files on the 490 minidisk.

3. Repeat substeps 1 on page 415 and 2 to copy the current HELP disks (19D, 19C, 19B...) to their corresponding alternate disks (49D, etc.) and to relabel the alternate disks as MNT49*n*. Check the minidisk definitions in your user directory to find the appropriate value for *endcyl*.

   Note: There is one HELP disk for each national language supported on your system. If you only support one national language, you only have to do this substep once.

What to Do Next

- To begin servicing CMS, see Chapter 10, "Receiving Service for CMS" on page 417.
- To begin servicing CP, see Chapter 11, "Receiving Service for CP" on page 423.
- To begin servicing DV, see Chapter 16, "Service to DV" on page 507.
- To begin servicing GCS, see Chapter 17, "Service to GCS" on page 517.

Chapter 10. Receiving Service for CMS

This chapter describes how to receive service from the PUT or COR tape for CMS.

Step 1. Preparation

Before you begin processing PUT or COR service, you must:

1. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cms ■
```

2. After you receive and apply service to CMS, you will want to keep the newly built CMS separate from the production CMS you are currently running. To do this, you must edit the DMSNGP ASSEMBLE file:

   Copy the latest version of the DMSNGP ASSEMBLE file to the 395 disk. (If there is already a copy of DMSNGP ASSEMBLE or DMSNGP TEXT on the 395 disk, rename it first.) Edit the copy and make the following changes:

   a. Change SYSDISK = 190 to SYSDISK = 490.

   b. Change IPLADDR = 190 to IPLADDR = 490.

   c. Change HELP = 19D to HELP = 49D (or HELP = 19C to HELP = 49C).

   d. Change SYSNAME = CMS to SYSNAME = *newname*, where *newname* is the name you choose for your test system. The sample procedure in Chapter 14, "Rebuilding CMS after Applying Service," uses ALTCMS.

   e. Change INSTSEG = CMSINST to INSTSEG = *newname*, where *newname* is the name you choose for your test installation segment. The sample procedure in Chapter 14, "Rebuilding CMS after Applying Service," uses ALTINST.

   f. Change the VERSION = and INSTID = parameters to identify your test system.

   g. Make sure that SAVESYS = NO. If SAVESYS = YES, when you IPL CMS the system will try to save the alternate CMS segment that you will define in Step 10 of Chapter 14, "Rebuilding CMS after Applying Service" in the existing CMS segment.

   These changes will cause the new CMS nucleus (with service) to be built on the 490 minidisk. (If you have applied service before and the nucleus you are using now is on the 490 minidisk, SYSDISK and IPLADDR are already 490. Change them back to 190 so that the new nucleus will be built on the 190 minidisk.)

3. You will probably want VMFREC to merge the DELTA and APPLY minidisk strings in accordance with the :MERGE. tag in the product parameter file. If you want to do it manually, or if you want to keep any disk string from being merged, see Appendix G, "Controlling Disk String Merges" on page 861.

4. Determine your system default national language:

```
query lang ■
langid
```

langid can be:

Langid	Language
AMENG	Mixed-case American English
KANJI	Kanji (Japan)
UCENG	Uppercase American English
PORTG	Brazilian Portuguese
FRANC	French
GER	German

If the system default national language is uppercase American English (UCENG), you must make these changes:

a. Change the :NLS. tags in the product parameter file from AMENG to UCENG.

b. Convert the following files to uppercase:

- $VMFMSG$ EXEC
- $VMFMSG$ $EXEC

```
xedit $vmfmsg$ {exec|$exec} ■
uppercas * ■
file ■
```

Note: If you receive service to these files, you must copy the changes in the replacement files to your modified version.

c. In the DMSNGP ASSEMBLE file, change the LANG parameter from AMENG to UCENG and the HELP disk address from 19D to 19C.

5. Rename the latest level of DMSNGP TEXT to save your old text file, then reassemble the DMSNGP ASSEMBLE file to include the changes you made in substeps 2 and 4:

```
vmfhasm dmsngp 56643089 cms ■
```

6. Copy the newly created DMSNGP TEXT file from your A-disk to your LOCAL1 disk and erase it from your A-disk:

```
copyfile dmsngp text a = = fm (olddate replace ■
erase dmsngp text a ■
```

fm is the LOCAL1 (395) disk.

Step 2. Receive Service

Note: If you are using ServiceLink to receive service electronically, use the procedure in Chapter 19, "Using ServiceLink to Receive Service" on page 533 to receive the service.

1. Attach a tape drive to MAINT:

   ```
   attach vdevno * 181 ■
   ```

2. Mount and ready the PUT or COR tape.

3. Map the PUT or COR tape and receive the documentation. Mapping the tape means verifying that all the necessary files exist. Enter:

   ```
   release c ■
   ```

   VMFREC INFO (MEMOS xxx loads documents to the C-disk if it is accessed, otherwise to the A-disk.

   ```
   vmfrec info (memos xxx ■
   DMSREC1852I This is n of n,
               level level {PUT|COR} tape
   ```

   xxx is **put** if you are mapping a PUT tape, or **cor** if you are mapping a COR tape. If the tape you mounted is not the type of tape you specified, you receive an error message.

   VMFREC INFO (MEMOS maps the tape and loads the tape map, PUT DOCUMENT or COR DOCUMENT, and the *Memo to Users* (56643089 MEMO) to your A-disk. Read these documents before going on. Print the documents to save a record of them. If you do not save a copy of these documents, they will be over-written the next time you receive a PUT or COR tape.

   If your system default language is uppercase American English, you may get message DMSMGM814E. This message is caused by a problem with the REXX date function. You can ignore it.

   ```
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

4. Review the receive exception log ($VMFREC $ERRLOG) to be sure that the MEMOs were loaded correctly:

   ```
   vmfview receive ■
   ```

5. Receive the service files to the target minidisks you specified in the product parameter file. Enter:

   ```
   vmfrec 56643089 cms (xxx ■
   DMSREC1852I This is n of n,
               level level {PUT|COR} tape
   ```

   xxx is **put** for a PUT tape, or **cor** for a COR tape. If you need to invoke an override file, substitute that file's name for **56643089**.

```
DMSREC1869R 56643089 CMS begins on {PUT|COR}
              level volume vol.  Mount
              correct volume and press ENTER
              or type QUIT.
■
DMSREC1852I This is n of n,
              level level {PUT|COR} tape
DMSREC1804I Receiving service for
              component CMS of product
              56643089
⋮
DMSREC1850I The processing for RESV by the
              VMFRCCOM EXEC was bypassed
Ready; T=n.nn/n.nn hh:mm:ss
```

If you do not have the correct volume mounted, mount it now.

The VMFREC EXEC loads the service files to the target minidisk.

6. Review the receive exception log ($VMFREC $ERRLOG), and correct any problems before continuing:

```
vmfview receive ■
Ready; T=n.nn/n.nn hh:mm:ss
```

To Receive More Than One COR Tape

If you want to receive additional COR tapes at this time:

1. You must suppress the automatic merging that occurs before a receive. See "Preventing Automatic Merging" in Appendix G, "Controlling Disk String Merges" on page 861, and choose a method of preventing automatic merging.
2. For each COR tape you want to receive, repeat substeps 2 – 6.
3. Create an apply list for the VMFAPPLY EXEC to use that contains the PTFs from all the COR tapes you just received:

```
listfile dmsxa $apc* fm ■
DMSXA    $APCALL  fm
DMSXA    $APC0109 fm
DMSXA    $APC0119 fm
DMSXA    $APC0126 fm
DMSXA    $APC9C22 fm
```

fm is the alternate DELTA disk (593).

```
xedit dmsxa $applist fm ■
get $apcymdd ■
file ■
```

Edit the DMSXA $APPLIST file on the alternate DELTA disk. This file will already contain the apply list from the last tape you loaded. Do a GET command in XEDIT once for each remaining DMSXA $APC*ymdd* file shown by your LISTFILE command.

Note: If the VMFAPPLY EXEC encounters duplicate entries for the same PTF, it issues a message that the PTF has already been applied. This is an informational message; you can ignore it.

4. Create an exclude list for the VMFAPPLY EXEC to use that contains the excluded PTFs from all the COR tapes you just received:

   Note: Do not be concerned if the exclude lists are empty.

```
listfile dmsxa $exc* fm ■
DMSXA    $EXCALL  fm
DMSXA    $EXC0109 fm
DMSXA    $EXC0119 fm
DMSXA    $EXC0126 fm
DMSXA    $EXC9C22 fm
```

   fm is the alternate DELTA disk (593).

```
xedit dmsxa $exclist fm ■
get $excymdd ■ file ■
```

   Edit the DMSXA $EXCLIST file on the alternate DELTA disk. This file will already contain the exclude list from the last tape you loaded. Do a GET command in XEDIT once for each remaining DMSXA $EXC*ymdd* file shown by your LISTFILE command.

End of To Receive More Than One COR Tape

What to Do Next

- If you are also servicing CP, you must receive the service for CP before applying the service to CMS. See Chapter 11, "Receiving Service for CP" on page 423.
- Otherwise, continue with service to CMS. See Chapter 12, "Applying Service to CMS" on page 427.

Chapter 11. Receiving Service for CP

This chapter describes how to receive service from the PUT or COR tape for CP.

Step 1. Preparation

Before you begin processing PUT or COR service, you must:

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. You will probably want VMFREC to merge the DELTA and APPLY minidisk strings in accordance with the :MERGE. tag in the product parameter file. If you want to do it manually, or if you want to keep any disk string from being merged, see Appendix G, "Controlling Disk String Merges" on page 861.

3. Determine your system default national language:

   Note: If you have already done this substep as part of receiving service for CMS, you do not need to do it again.

   ```
   query lang ■
   langid
   ```

   langid can be:

Langid	Language
AMENG	Mixed-case American English
KANJI	Kanji (Japan)
UCENG	Uppercase American English
PORTG	Brazilian Portuguese
FRANC	French
GER	German

   If the system default national language is uppercase American English (UCENG), you **must** make these changes:

   a. Change the :NLS. tags in the product parameter file from AMENG to UCENG.

   b. Convert the following files to uppercase:

      - $VMFMSG$ EXEC
      - $VMFMSG$ $EXEC

      ```
      xedit $vmfmsg$ {EXEC|$EXEC} ■
      uppercas * ■
      file ■
      ```

      Note: If you receive service to these files, you must copy the changes in the replacement files to your modified version.

Step 2. Receive Service

Note: If you are using ServiceLink to receive service electronically, use the procedure in Chapter 19, "Using ServiceLink to Receive Service" on page 533.

1. Attach a tape drive to MAINT:

   ```
   attach vdevno * 181 ■
   ```

2. Mount and ready the PUT or COR tape.

3. Map the PUT or COR tape and receive the documentation. Mapping the tape means verifying that all the necessary files exist. Enter:

   ```
   release c ■
   ```

   VMFREC INFO (MEMOS *xxx* loads documents to the C-disk if it is accessed, otherwise to the A-disk.

   ```
   vmfrec info (memos xxx ■
   DMSREC1852I This is n of n,
               level level {PUT|COR} tape
   ```

   xxx is **put** if you are mapping a PUT tape, or **cor** if you are mapping a COR tape. If the tape you mounted is not the type of tape you specified, you receive an error message.

   VMFREC INFO (MEMOS maps the tape and loads the tape map, PUT DOCUMENT or COR DOCUMENT, and the *Memo to Users* (56643089 MEMO) to your A-disk. Read these documents before going on. Print the documents to save a record of them. If you do not save a copy of these documents, they will be over-written the next time you receive a PUT or COR tape.

   If your system default language is uppercase American English, you may get message DMSMGM814E. This message is caused by a problem with the REXX date function. You can ignore it.

   ```
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

4. Review the receive exception log ($VMFREC $ERRLOG) to be sure that the MEMOs were loaded correctly:

   ```
   vmfview receive ■
   ```

5. Receive the service files to the target minidisks you specified in the product parameter file. Enter:

   ```
   vmfrec 56643089 cp (xxx ■
   DMSREC1852I This is n of n,
               level level {PUT|COR} tape
   ```

   xxx is **put** for a PUT tape, or **cor** for a COR tape. If you need to invoke an override file, substitute that file's name for **56643089**.

```
DMSREC1869R 56643089 CP begins on {PUT|COR}
              level volume vol.  Mount
              correct volume and press ENTER
              or type QUIT.
■
DMSREC1852I This is n of n,
             level level {PUT|COR} tape
DMSREC1804I Receiving service for
             component CP of product
             56643089
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

If you do not have the correct volume mounted, mount it now.

The VMFREC EXEC loads the service files to the target minidisk.

6. Review the receive exception log ($VMFREC $ERRLOG), and correct any problems before continuing:

```
vmfview receive ■
Ready; T=n.nn/n.nn hh:mm:ss
```

To Receive More Than One COR Tape

If you want to receive additional COR tapes at this time:

1. You must suppress the automatic merging that occurs before a receive. Read "Preventing Automatic Merging" on page 861 in Appendix G, "Controlling Disk String Merges" on page 861 and choose a method of preventing automatic merging.
2. For each COR tape you want to receive, repeat substeps 2 – 6.
3. Create an apply list for the VMFAPPLY EXEC to use that contains the PTFs from all the COR tapes you just received:

```
listfile hcpxa $apc* fm ■
HCPXA    $APCALL  fm
HCPXA    $APC0109 fm
HCPXA    $APC0119 fm
HCPXA    $APC0126 fm
HCPXA    $APC9C22 fm
```

fm is the alternate DELTA disk (593).

```
xedit hcpxa $applist fm ■
get $apcymdd ■
file ■
```

Edit the HCPXA $APPLIST file on the alternate DELTA disk. This file will already contain the apply list from the last tape that you loaded. Do a GET command in XEDIT once for each remaining HCPXA $APC*ymdd* file shown by your LISTFILE command.

Note: If the VMFAPPLY EXEC encounters duplicate entries for the same PTF, it issues a message telling you that the PTF has already been applied. This is an informational message; you can ignore it.

4. Create an exclude list for the VMFAPPLY EXEC to use that contains the excluded PTFs from all the COR tapes you just received:

Note: Do not be concerned if the exclude lists are empty.

```
listfile hcpxa $exc* fm ■
HCPXA    $EXCALL  fm
HCPXA    $EXC0109 fm
HCPXA    $EXC0119 fm
HCPXA    $EXC0126 fm
HC!XA    $EXC9C22 fm
```

fm is the alternate DELTA disk (593).

```
xedit hcpxa $exclist fm ■
get $excymdd ■
file ■
```

Edit the HCPXA $EXCLIST file on the alternate DELTA disk. This file will already contain the exclude list from the last tape that you loaded. Do a GET command in XEDIT once for each remaining HCPXA $EXC*ymdd* file shown by your LISTFILE command.

End of To Receive More Than One COR Tape

What to Do Next

- If you are also servicing CMS, you must apply the service for CMS before you apply service to CP. See Chapter 12, "Applying Service to CMS" on page 427.
- Otherwise, continue with service to CP. See Chapter 13, "Applying Service to CP" on page 429.

Chapter 12. Applying Service to CMS

This chapter describes how to apply service from the PUT tape or COR tape to CMS.

Warning: If you are servicing both CP and CMS, you **must** receive both CP and CMS service before applying service to CMS, because you need CP macro libraries to apply service to CMS. Some of these libraries may have been serviced.

Should You Be Doing This Now?

- If you have not yet received service for CMS, see Chapter 10, "Receiving Service for CMS" on page 417.
- If you are also servicing CP, you must receive the service for CP before applying the service to CMS. If you have not yet received service for CP, see Chapter 11, "Receiving Service for CP" on page 423.
- Otherwise, continue with this chapter.

1. Consult your IBM service representative for the appropriate preventive service planning (PSP) bucket for your system's VMSUP level and PUT level. Check the bucket and add the appropriate entries to the exclude list. The exclude list is called DMSXA $EXCLIST.

   Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prerequisite for the second PTF.

   If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

2. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

3. Apply the service, that is, create auxiliary control files. Enter:

   ```
   vmfapply 56643089 cms (xxx ■
   DMSAPP1853I Processing PTF ptfnum
   ⋮
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   xxx is **put** for PUT service, or **cor** for COR service.

4. Review the apply exception log ($VMFAPP $ERRLOG), and correct any problems before continuing. The information in the apply exception log is used later when you rebuild CMS.

`vmfview apply` ■

If you receive message number DMSAPP1885W, VMFAPPLY has applied service to a module for which you have a local modification.

You can use the PF keys with the VMFVIEW EXEC to get useful lists of information. If you position the cursor on a line that contains message 1885W and press **PF2**, VMFVIEW displays all the occurrences of message 1885W. Similarly, if you press **PF6**, VMFVIEW displays all the messages with the BD: prefix. You can use the XEDIT PUT command to save these messages in a file for future reference.

Decide what you want to do with the local modification:

- To remove a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543, and Chapter 22, "Removing Service from VM/XA SP" on page 559.
- To rework a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543.
- To keep a local modification, do nothing now. You will regenerate the part to pick up the local modification when you rebuild CMS.

Save the names of all the parts for which you have both local modifications and IBM service. You will need the part names later when you rebuild CMS.

Whether you remove, rework, or keep local modifications, you do not have to reissue VMFAPPLY.

What to Do Next

- If you are also servicing CP, you must apply the service for CP before you rebuild CMS. See Chapter 13, "Applying Service to CP" on page 429.
- Otherwise, continue with service to CMS. See Chapter 14, "Rebuilding CMS after Applying Service" on page 431.

Chapter 13. Applying Service to CP

This chapter describes how to apply service from the PUT or COR tape to CP.

Should You Be Doing This Now?

- If you are also servicing CMS, you must apply the service to CMS before applying the service to CP. If you have not yet applied service to CMS, see Chapter 12, "Applying Service to CMS" on page 427.
- Otherwise, continue with this chapter.

1. Consult your IBM service representative for the appropriate preventive service planning (PSP) bucket for your system's VMSUP level and PUT level. Check the bucket and add the appropriate entries to the exclude list. The exclude list is called HCPXA $EXCLIST.

   Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prerequisite for the second PTF.

   If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

2. Establish the proper minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

3. Apply the service, that is, create auxiliary control files. Enter:

   ```
   vmfapply 56643089 cp (xxx ■
   DMSAPP1853I Processing PTF ptfnum
   :
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   xxx is **put** for PUT service, or **cor** for COR service.

4. Review the apply exception log ($VMFAPP $ERRLOG), and correct any problems before continuing. The information in the apply exception log is used later when you rebuild CP.

   ```
   vmfview apply ■
   ```

   If you receive message number DMSAPP1885W, VMFAPPLY has applied service to a module for which you have a local modification.

   You can use the PF keys with the VMFVIEW EXEC to get useful lists of information. If you position the cursor on a line that contains message 1885W and press **PF2**, VMFVIEW displays all the occurrences of message 1885W. Similarly, if you press **PF6**, VMFVIEW displays all the messages with the BD: prefix. You can use the XEDIT PUT command to save these messages in a file for future reference.

Decide what you want to do with the local modification:

- To remove a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543, and Chapter 22, "Removing Service from VM/XA SP" on page 559.
- To rework a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543.
- To keep a local modification, do nothing now. You will regenerate the part to pick up the local modification when you rebuild CP.

Save the names of all the parts for which you have both local modifications and IBM service. You will need the part names later when you rebuild CP.

Whether you remove, rework, or keep local modifications, you do not have to reissue VMFAPPLY.

What to Do Next

- If you are also servicing CMS, you must rebuild CMS before you rebuild CP. See Chapter 14, "Rebuilding CMS after Applying Service" on page 431.
- Otherwise, continue with service to CP. See Chapter 15, "Rebuilding CP after Applying Service" on page 487.

Chapter 14. Rebuilding CMS after Applying Service

This chapter describes a step-by-step procedure for rebuilding CMS after applying program update service, corrective service, or local service.

Should You Be Doing This Now?

- If you are also servicing CP, you must apply the service to CP before rebuilding CMS. See Chapter 13, "Applying Service to CP" on page 429.
- Otherwise, continue with this chapter.

Step 1. Build a New CMS Macro Library

> **Note:** This step describes a method of updating the CMS macro libraries that isolates all of the serviced macros and control blocks into a separate MACLIB using the VMFMAC EXEC. The new MACLIB is added to the MACS record in the DMSMXA CNTRL file.
>
> An alternate method of updating the CMS macro libraries is to update each affected MACLIB with the changed macro or control block by using the MACLIB command. (See *VM/XA SP CMS Command Reference* for details on the MACLIB command.)
>
> If you choose the alternate method of building new CMS macro libraries, do so now and continue with "Step 2. Build a New CP Macro Library" on page 435.

1. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cms ■
```

2. Determine the macros and control blocks you have service for:
   - Search the DELTA string for all macros and control blocks you have service for:

```
listfile * E*DS fm1 (exec args fname ■
```
fm1 is the alternate DELTA (593) disk.

```
listfile * E*DS fm2 (append args fname ■
```
fm2 is the intermediate alternate DELTA (691) disk.

```
listfile * E*DS fm3 (append args fname ■
```
fm3 is the current DELTA (293) disk.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.
   - Save the filenames in the CMSUPDAT EXEC and erase the CMS EXEC:

```
copyfile cms exec a cmsupdat = = (replace ■
erase cms exec a ■
```

   - Determine which of the updated files are macros (filetype MACRO), and which are control blocks (filetype COPY). Store the results in the CMSNEW EXEC:

```
cmsupdat listfile % macro * (append args ftype ■
cmsupdat listfile % copy * (append args ftype ■
```
Ignore the messages that appear on your screen while these two commands are processing.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.

- Save the filenames in the CMSNEW EXEC:

```
copyfile cms exec a cmsnew = = (replace ■
```

- Edit the CMSNEW EXEC to remove any duplicate entries:

```
xedit cmsnew exec ■
sort * 1 15 ■
:
file ■
```

3. Determine the macros and control blocks for which you have local service. Follow your installation's naming conventions for locally serviced files. Locally serviced files should reside on the LOCAL1 disk string.
   - If no macros and control blocks have local service, continue with substep 4.
   - Otherwise, add your locally serviced macros and control blocks to CMSNEW EXEC. When you are done, the CMSNEW EXEC should list all macros and control blocks for which you have received IBM service (if any), plus any macros and control blocks for which you have local service.

     If you do not have a CMSNEW EXEC on your A-disk, copy the DMSGPI EXEC to use as a template:

```
copyfile dmsgpi exec fm cmsnew = a (recfm f lrecl 80 ■
```

     If you created the CMSNEW EXEC by copying the DMSGPI EXEC, use the contents of the DMSGPI EXEC only for a guide to the format of the entries. After you have added the macro and control block names you need, in the proper format, delete the macro names that originated from the DMSGPI EXEC:

```
xedit cmsnew exec a ■
:
file ■
```

4. If you have no updated macros or control files at this point, continue with "Step 2. Build a New CP Macro Library" on page 435.
5. Generate the new macro library on your A-disk (191) by using the VMFMAC EXEC:

```
vmfmac cmsnew dmsmxa ■
DMSUPD178I Updating macroname MACRO fm
DMSUPD178I Applying macroname update fm
macroname MACRO ADDED
:
Ready; T=n.nn/n.nn hh:mm:ss
```

   DMSMXA CNTRL is a special control file used for generating CMS macro libraries. Do not confuse it with DMSXA CNTRL, the CMS main control file.

6. The contents of CMSNEW COPY were appended to CMSNEW MACLIB. You can now erase CMSNEW COPY (a file created by VMFMAC) on your A-disk (191):

```
erase cmsnew copy a ■
```

7. Add the new macro library name to the TEXT MACS card in the DMSXA CNTRL file. (DMSXA CNTRL is the the CMS main control file. Do not confuse it with DMSMXA CNTRL, used for generating CMS macro libraries.)

   List the new macro library first so that you use the updated macros. If you have already done this during previous service to CMS, you do not have to do this now.

   ```
   xedit dmsxa cntrl fm ■
   locate/TEXT MACS ■
   TEXT MACS maclib ...
   change/MACS/MACS CMSNEW/ ■
   TEXT MACS CMSNEW maclib ...
   file ■
   ```

   fm is the CMS alternate BUILD1 disk (490).

8. Copy the changed files from your A-disk to the CMS database: *fm* is the alternate DELTA disk (593) if you have **no** local service to the macros or control blocks. *fm* is the alternate LOCAL disk (395) if you **have** any local service to the macros or control blocks.

   ```
   copyfile cmsnew exec a = = fm (olddate replace ■
   copyfile cmsnew maclib a = = fm (olddate replace ■
   ```

9. To make the MACLIB you just created available to general users, copy it from the CMS database to your CMS alternate BUILD1 disk (490):

   ```
   copyfile cmsnew maclib origin-fm = = target-fm2 (olddate replace ■
   ```

   origin-fm is the disk you copied the changed files to in the previous substep. *target-fm* is the CMS alternate BUILD1 disk (490).

10. Erase the working copies on your A-disk:

    ```
    erase cmsnew exec a ■
    erase cmsupdat exec a ■
    erase cmsnew maclib a ■
    ```

Step 2. Build a New CP Macro Library

> **Note:** This step describes a method of updating the CP macro libraries that isolates all of the serviced macros and control blocks into a separate MACLIB using the VMFMAC EXEC. The new MACLIB is added to the MACS record in the HCPXA CNTRL file.
>
> An alternate method of updating the CP macro libraries is to update each affected MACLIB with the changed macro or control block by using the MACLIB command. (See *VM/XA SP CMS Command Reference* for details on the MACLIB command.)
>
> If you choose the alternate method of building new CP macro libraries, do so now and continue with "Step 3. Update the CMS Message Repository" on page 438.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Determine the macros and control blocks you have service for:

   - Search the DELTA string for all macros and control blocks you have service for:

`listfile * A*HP` *fm1* `(execs args fname ■`	*fm1* is the alternate DELTA (594) disk.
`listfile * A*HP` *fm2* `(append args fname ■`	*fm2* is the intermediate alternate DELTA (591) disk.
`listfile * A*HP` *fm3* `(append args fname ■`	*fm3* is the current DELTA (294) disk.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.
   - Save the filenames in the CPUPDATE EXEC and erase the CMS EXEC:

     ```
     copyfile cms exec a cpupdate = = (replace ■
     erase cms exec a ■
     ```

   - Determine which of the updated files are macros (filetype MACRO), and which are control blocks (filetype COPY). Store the results in the CPNEW EXEC:

     ```
     cpupdate listfile % macro * (append args ftype ■
     cpupdate listfile % copy * (append args ftype ■
     ```

     Ignore the messages that appear on your screen while these two commands are processing.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.

- Save the filenames in the CPNEW EXEC:

```
copyfile cms exec a cpnew = = (replace ■
```

- Edit the CPNEW EXEC to remove any duplicate entries:

```
xedit cpnew exec ■
sort * 1 15 ■
:
file ■
```

3. Determine the macros and control blocks for which you have local service. Follow your installation's naming conventions for locally serviced files. Locally serviced files should reside on the LOCAL1 disk string.
   - If no macros and control blocks have local service, continue with substep 4.
   - Otherwise, add your locally serviced macros and control blocks to CPNEW EXEC. When you are done, the CPNEW EXEC should list all macros and control blocks for which you have received IBM service (if any), plus any macros and control blocks for which you have local service.

     If you do not have a CPNEW EXEC on your A-disk, copy the HCPGPI EXEC to use as a template:

```
copyfile hcpgpi exec fm cpnew = a (recfm f lrecl 80 ■
```

     If you created the CPNEW EXEC by copying the HCPGPI EXEC, use the contents of the HCPGPI EXEC only for a guide to the format of the entries. After you have added the macro and control block names you need, in the proper format, delete the macro names that originated from the HCPGPI EXEC:

```
xedit cpnew exec a ■
:
file ■
```

4. If you have no updated macros or control files at this point, continue with "Step 3. Update the CMS Message Repository" on page 438.
5. Generate the new macro library on your A-disk (191) by using the VMFMAC EXEC:

```
vmfmac cpnew hcpmxa ■
DMSUPD178I Updating macroname MACRO fm
DMSUPD178I Applying macroname update fm
macroname MACRO ADDED
:
Ready; T=n.nn/n.nn hh:mm:ss
```

6. The contents of CPNEW COPY were appended to CPNEW MACLIB. You can now erase CPNEW COPY (a file created by VMFMAC) on your A-disk (191):

```
erase cpnew copy a ■
```

7. Add the new macro library name to the TEXT MACS card in the HCPXA CNTRL file.

   List the new macro library first so that you use the updated macros. If you have already done this during previous service to CP, you do not have to do this now.

   ```
   xedit hcpxa cntrl fm ■
   locate/TEXT MACS ■
   TEXT MACS maclib ...
   change/MACS/MACS CPNEW/ ■
   TEXT MACS CPNEW maclib ...
   file ■
   ```

   fm is the CP BASE1 disk (194).

8. Copy the changed files from your A-disk to the CP database: *fm* is the alternate DELTA disk (594) if you have **no** local service to the macro or copy files. *fm* is the alternate LOCAL disk (295) if you **have** any local service to the macro or copy files.

   ```
   copyfile cpnew exec a = = fm (olddate replace ■
   copyfile cpnew maclib a = = fm (olddate replace ■
   ```

9. To make the MACLIB you just created available to general users, copy it from the CP database to your CMS alternate BUILD1 disk (490):

   ```
   copyfile cpnew maclib origin-fm = = target-fm2 (olddate replace ■
   ```

   origin-fm is the disk you copied the changed files to in the previous substep. *target-fm* is the CMS alternate BUILD1 disk (490).

10. Erase the working copies on your A-disk:

    ```
    erase cpnew exec a ■
    erase cpupdate exec a ■
    erase cpnew maclib a ■
    ```

Step 3. Update the CMS Message Repository

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

2. Determine your system default national language:

   ```
   query lang ■
   ```

   langid — See Table 5 on page 311 to identify the language corresponding to *langid*. Note the country code for that language.

3. Invoke the VMFNLS EXEC to update the CMS message repository:

   ```
   vmfnls dmsmesy repos 56643089 cms ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   y is the country code for your system national language.

4. If you have updates to the message repositories for any other national languages installed on your system, repeat substep 3 for each language.

5. Copy the changed files from your A-disk to the CMS database: *fm* is the alternate DELTA disk (593) if you have **no** local service to the macro or copy files. *fm* is the alternate LOCAL disk (395) if you **have** any local service to the macro or copy files.

   ```
   copyfile dmsmesy {text|txtnnnnn} a = = fm (olddate replace ■
   copyfile dmsmesy listing a = = fm (olddate replace ■
   ```

6. Erase the working copies on your A-disk:

   ```
   erase dmsmesy {text|txtnnnnn} a ■
   erase dmsmesy listing a ■
   ```

Step 4. Assemble the Changed ASSEMBLE Files

Do this step if you have local modifications to ASSEMBLE files.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

2. Use the list of parts for which you have both local modifications and IBM service that you made in Chapter 2, "Applying Service to CMS." You must reassemble any such ASSEMBLE files to pick up the local updates.

3. Copy and unpack the ASSEMBLE files you need to reassemble from the CMS current BASE2 disk (393) to your A-disk:

   ```
   copy dmsxxx assemble fm = = a (unpack olddate ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   Substitute the last 3 characters of the filename of the ASSEMBLE file for *xxx*.

4. Use the VMFHASM EXEC to update and assemble all the files for which you have both IBM service and local modifications:

   ```
   vmfhasm dmsxxx 56643089 cms ■
   DMSUPD178I UPDATING DMSxxx ASSEMBLE filemode
   DMSUPD178I APPLYING DMSxxx filetype filemode
   ASSEMBLING DMSxxx
   DMSxxx {TEXT|xxxnnnnn} CREATED
   PRT FILE nnnn SENT FROM MAINT PRT AS nnnn
            RECS nnnn COPY 001 I NOHOLD NOKEEP
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   Substitute the last 3 characters of the filename of the ASSEMBLE file for *xxx*.

   If you assemble DMSNUC ASSEMBLE, you may receive some assembler warning messages and a return code of 4. The warnings and return code are normal: you can ignore them. However, the DMSNUC text deck you create is valid.

5. Copy all the text files from the A-disk to the alternate LOCAL1 (395) disk:

   ```
   copy dmsxxx {text|xxxnnnnn} a = = fm (replace olddate ■
   ```

6. If you reassembled any files that are not in the CMS load list (CMSLOAD EXEC), copy the text decks to the alternate LOCAL1 disk again, giving the copy the filetype TEXT:

   ```
   copy dmsxxx {text|xxxnnnnn} a = text fm (replace olddate ■
   ```

7. Erase all the ASSEMBLE files and text files on your A-disk:

   ```
   erase dmsxxx assemble a ■
   erase dmsxxx {text|xxxnnnnn} a ■
   ```

8. Repeat substeps 3 through 7 for each file that must be reassembled.

Step 5. Build the New Product Parameter File

Do this step if the 56643089 $PPF file has been serviced.

1. Establish the correct minidisk access order:

```
vmfsetup 56643089 cms ■
DMSACC724I vdevno replaces fm (vdevno
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Look for the latest version of the product parameter file:

```
filelist 56643089 $P* * ■
```

Examine each version of the product parameter file. The product parameter file has an update history at the top. The latest level is the one with the most updates. Use the update history, **not the date and time stamp,** to determine which is the most recent version.

3. Copy the latest level of the product parameter file to the TASK disk (5E5):

```
copyfile / = $ppf fm (olddate replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Issue this command from the FILELIST screen, on the line listing the file you want to copy. *fm* is the TASK disk (5E5).

Step 6. Build the CMS Nucleus

> **Warning: Before you run VMFBLD, you** must **rebuild the CMS build lists if they have been serviced. The complete list of CMS build lists is found in the :BLD. section of the CMS portion of the PPF.**
>
> **Use the procedures in "Step 8. Regenerate System Product Interpreter Programs" on page 449 to rebuild the build lists, and return here.**

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Generate the new CMS nucleus:

```
vmfbld 56643089 cms (punch ■
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
⋮
```

This invokes the VMFBLD EXEC to build a new nucleus for CMS. The VMFBLD EXEC performs a number of system generation functions for you. For more information on this EXEC, see "VMFBLD EXEC" on page 652.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 N NOHOLD NOKEEP
```

The system loader punches the CMS nucleus. This message tells you that the punch file is complete.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in the virtual reader:

```
query rdr * all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class * issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. IPL your virtual reader:

```
ipl 00c ■
DMSINS327I  The installation saved segment could not be loaded
```

Informational message: The CMSINST installation segment has not been loaded and saved yet.

```
VM/XA ALTCMS mm/dd/yy
■
```

The version identification you defined appears here and each time that you IPL 490 or IPL CMS.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

The ready message indicates that the nucleus has been loaded to your 490 minidisk successfully.

9. Receive the CMS load map into your reader:

```
spool prt * nohold ■
close prt * ■
```

10. Query the reader to identify the CMS load map:

```
query rdr * all ■
ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,100 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

```
vmfsetup 56643089 cms ■
receive fileno fn ft fm ■
fn ft fm1 created
DMSRDC738I Record length is 132 bytes
File fn ft fm received from MAINT at
  nodeid as fn ft fm
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS load map is loaded on the alternate LOCAL1 (395) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1–9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

11. Review the CMS load map to verify that any local modifications to CMS modules have been applied.

Note: From this point on:

- To IPL your **production system,** issue:
  ipl 190 ■
- To IPL your **test system,** issue:
  ipl 490 ■

Step 7. Generate Executable Modules

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

2. Regenerate any MODULEs and LOADLIBs for which you have received IBM service:

   - Make and sort a list of the files you are servicing:

     ```
     listfile * text fm (exec ■
     ```
     fm is the alternate DELTA disk (593).

     ```
     listfile * text fm (append ■
     ```
     fm is the LOCAL1 disk (395).

     ```
     rename cms exec a dmstext = = ■
     xedit dmstext exec ■
     sort * 1 15 ■
     file ■
     ```
     The **exec** option saves the list in a file called CMS EXEC. Any previously existing CMS EXEC on your A-disk is erased.

   - Check this list against Table 12. You must generate an executable module for any file you find listed there.
   - Find each executable module you have to regenerate in the first column of the table. Enter the command in the second column. Make sure that the file named in the third column is created on your A-disk. You can ignore any `INVALID CARD` messages you receive while running CMSGEND.

     During regeneration of the modules in Table 12, files with a filetype of LKEDIT may be created and erased. You do not need them.

Table 12 (Page 1 of 4). CMSGEND Files

Assemble Filename	Command to Enter	File Created by This Command
Note: If you receive service for a CMS text deck not listed in this table, it will be regenerated in some other way. You do not need to regenerate it now. (See Table 15 on page 567 in Appendix A, "VM/XA System Product Regeneration Requirements" for the procedure used. Text decks not listed in this table, or in Table 15 on page 567, become part of the nucleus when the system is built if they are listed in the CMS load list.)		
DMSAMS	**cmsgend amserv ■**	AMSERV MODULE
DMSASD DMSASM	**cmsgend assemble ■**	ASSEMBLE MODULE (see Note 5 on page 447)
DMSASN	**cmsgend assgn ■**	ASSGN MODULE
DMSBCT DMSBLG DMSCDI DMSDFT DMSSMG DMSUSR	**cmsgend dmsdft ■**	DMSDFT MODULE

Table 12 (Page 2 of 4). CMSGEND Files

Assemble Filename	Command to Enter	File Created by This Command
DMSBOF DMSBUS DMSCCM DMSCIA DMSICT DMSPBS DMSSAP DMSSNC DMSSUP DMSTRC	**cmsgend dmscut** ■	DMSCUT MODULE
DMSBTB	**cmsgend cmsbatch** ■	CMSBATCH MODULE
DMSCCK	**cmsgend catcheck** ■	CATCHECK MODULE
DMSCMP	**cmsgend compare** ■	COMPARE MODULE
DMSDLK	**cmsgend doslked** ■	DOSLKED MODULE
DMSDSK	**cmsgend disk** ■	DISK MODULE
DMSDSL	**cmsgend doslib** ■	DOSLIB MODULE
DMSDSV	**cmsgend dserv** ■	DSERV MODULE
DMSEDC DMSEDF DMSEDI DMSEDX DMSGIO DMSSCR DMSZIT	**cmsgend edit** ■	EDIT MODULE (see Note 1 on page 447)
DMSEXG	**cmsgend dcssgen** ■	DCSSGEN MODULE
DMSEXM	**cmsgend execmap** ■	EXECMAP MODULE
DMSFOR	**cmsgend format** ■	FORMAT MODULE
DMSGLB	**cmsgend global** ■	GLOBAL MODULE
DMSGND	**cmsgend gendirt** ■	GENDIRT MODULE
DMSHLB DMSHLD DMSHLI DMSHLP DMSHLS	**cmsgend helpconv** ■	HELPCONV MODULE
DMSICP IOPC*xxxx* IOPP*xxxx*	**cmsgend iocp** ■	IOCP MODULE
DMSIMA	**cmsgend imagemod** ■	IMAGEMOD MODULE
DMSLBD	**cmsgend labeldef** ■	LABELDEF MODULE
DMSLBM	**cmsgend maclib** ■	MACLIB MODULE
DMSLBT	**cmsgend txtlib** ■	TXTLIB MODULE

Table 12 (Page 3 of 4). CMSGEND Files

Assemble Filename	Command to Enter	File Created by This Command
DMSLDS	cmsgend listds ■	LISTDS MODULE
DMSLLU	cmsgend listio ■	LISTIO MODULE
DMSLMX	cmsgend tape ■	TAPE MODULE (see Note 2 on page 447)
DMSMDP	cmsgend modmap ■	MODMAP MODULE
DMSMGC DMSMGD DMSMGE	cmsgend genmsg ■	GENMSG MODULE
DMSMIG	cmsgend maclmig ■	MACLMIG MODULE
DMSMVE	cmsgend movefile ■	MOVEFILE MODULE
DMSNXD	cmsgend nucxdrop ■	NUCXDROP MODULE
DMSOPT	cmsgend option ■	OPTION MODULE
DMSOSR	cmsgend osrun ■	OSRUN MODULE
DMSOVR	cmsgend svctrace ■	SVCTRACE MODULE
DMSOVS	cmsgend dmsovs ■	DMSOVS MODULE
DMSPCA DMSPCB DMSPCC DMSPCR DMSPCT DMSPCW	cmsgend dmspcc ■	DMSPCC MODULE
DMSPOA DMSPOC DMSPOD DMSPOE DMSPOL DMSPON DMSPOP DMSPOQ DMSPOR DMSPOS	cmsgend prop ■	PROPLIB LOADLIB (see Note 3 on page 447)
DMSPRE	cmsgend preload ■	PRELOAD MODULE
DMSPRV	cmsgend pserv ■	PSERV MODULE
DMSPUN	cmsgend punch ■	PUNCH MODULE
DMSRDC	cmsgend readcard ■	READCARD MODULE
DMSRDR	cmsgend rdr ■	RDR MODULE
DMSRNE	cmsgend renum ■	RENUM MODULE
DMSRRV	cmsgend rserv ■	RSERV MODULE
DMSRSV	cmsgend reserve ■	RESERVE MODULE
DMSSFD	cmsgend savefd ■	SAVEFD MODULE

Table 12 (Page 4 of 4). CMSGEND Files

Assemble Filename	Command to Enter	File Created by This Command
DMSSPR	cmsgend setprt ■	SETPRT MODULE
DMSSRT	cmsgend sort ■	SORT MODULE
DMSSRV	cmsgend sserv ■	SSERV MODULE
DMSSSK	cmsgend setkey ■	SETKEY MODULE
DMSSYN	cmsgend synonym ■	SYNONYM MODULE
DMSTMA	cmsgend tapemac ■	TAPEMAC MODULE
DMSTPD	cmsgend tappds ■	TAPPDS MODULE
DMSTPE DMSTPF DMSTPG DMSTPH DMSTPI DMSTPJ	cmsgend tape ■	TAPE MODULE (see Note 2 on page 447)
DMSTYP	cmsgend type ■	TYPE MODULE
DMSUPD	cmsgend update ■	UPDATE MODULE
DMSUTL	cmsgend loadlib ■	LOADLIB MODULE
DMSXMS	cmsgend dmsxms ■	DMSXMS MODULE (see Note 4 on page 447)
DMSXRE	cmsgend dmsxre ■	DMSXRE MODULE (see Note 4 on page 447)
DMSZAP	cmsgend zap ■	ZAP MODULE
VMFCLEAR	load vmfclear (origin trans nomap type ■ genmod vmfclear module a (nomap system nostr all ■	VMFCLEAR MODULE
VMFDATE	cmsgend vmfdate ■	VMFDATE MODULE
VMFDOS	cmsgend vmfdos ■	VMFDOS MODULE
VMFLOAD	cmsgend vmfload ■	VMFLOAD MODULE

Notes:

1. When the CMSGEND EXEC procedure is invoked for EDIT, it creates the EDIT module. Then it automatically reinvokes itself to create the EDMAIN module.
2. When the CMSGEND EXEC is invoked for TAPE, it creates the TAPE module and then reinvokes itself to create the DMSLMX and DMSTP*x* modules.
3. You get messages DMSSLK0008W and DMSSOP036E when you regenerate PROPLIB. You can ignore them.
4. All EDIT source files, except DMSXMS and DMSXRE, are contained within the CMS nucleus.
5. The assembler text decks normally reside on the system S-disk as filemode S1. This disk must be accessed as some additional filemode before you issue the CMSGEND ASSEMBLE command in order to locate these files. Refer to "CMSGEND EXEC" on page 577.

3. Test the modules.
4. Copy any MODULEs and LOADLIBs generated on the A-disk to the alternate DELTA disk. Then erase them on the A-disk:

```
copyfile fn MODULE|LOADLIB A = = fm ■
erase fn MODULE|LOADLIB A ■
```

fm is the alternate DELTA disk (593).

5. Copy all serviced MODULEs to their appropriate build disks:
   - Create a list of the MODULEs you are servicing.

     Note: This list includes all serviced MODULEs on the alternate DELTA disk whether or not you regenerated them in the previous step.

     ```
     filelist * MODULE fm ■
     ```

     fm is the alternate DELTA disk (593).

   - Create a list of all parts for each filename in the filelist by pressing **PF9**. Press **PF9** for each file in the original filelist.
   - Copy the new executable version of the module to the appropriate build disks:
     - If an executable version (with filetype MODULE) exists on the BUILD1 disk, copy the new version there.
     - If an executable version exists on the TASK disk, copy the new version there.
     - If an executable version does not reside on one of these disks, refer to the documentation supplied with the PTF to determine where it should reside.

     ```
     copyfile / = = fm2 (olddate replace ■
     ```

     Enter this command on the filelist screen next to the file you want to copy. You are copying these files from the alternate DELTA disk (593) to the BUILD1 disk (490), the TASK disk (5E5), or both.

     Use filemode 2 for files copied to the BUILD1 disk so they can be seen by other users.

   - Press **PF2** to refresh the screen to verify that your changes have been made.
   - When you have processed each file in the original filelist screen, use **PF3** to exit filelist.
6. Repeat substep 5 for LOADLIBs.

Step 8. Regenerate System Product Interpreter Programs

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

2. Regenerate any EXECs and XEDIT macros for which you have received IBM service and for which you have local modifications. (If you have none, go to substep 3).

   - Use the list of parts for which you have both local modifications and IBM service that you made in Chapter 12, Applying Service to CMS.
   - If you reworked or kept your local modifications, you must now regenerate the EXECs and XEDIT macros. Repeat the following three commands for each file that needs to be regenerated.

     Note: *ft* is either EXEC or XEDIT. Remember to include the leading dollar sign ($) when it is specified.

copyfile *fn* **$***ft origin-fm* = = *destination-fm* ■	*origin-fm* is the BASE2 disk (393). *destination-fm* is the alternate DELTA disk (593). The file is copied to the alternate DELTA disk because the EXECUPDT command puts the object file on the same disk as the source file.
execupdt *fn ft fm* **(ctl dmsxa hist** *option* ■	DMSXA is the CMS control filename. For any files with a SID code in columns 63 – 71, use the SID option. The following files do not include SID codes: PROPEPIF EXEC PROPHCHK EXEC PROPLGER EXEC PROPPCHK EXEC PROPPROF EXEC PROPRTCV EXEC PROPST EXEC.
erase *fn* **$***ft fm* ■	*fm* is the alternate DELTA disk (593).

3. Copy all serviced EXECs to their appropriate build disks:

   - Create a list of the EXECs you are servicing.

     Note: This list includes all the serviced EXECs on your alternate DELTA disk whether or not you regenerated them in the previous step.

filelist * EXEC *fm* ■	*fm* is the alternate DELTA disk (593).

   - Create a list of all parts of each filename in the filelist using **PF9**. Press **PF9** for each file in the original filelist.

- Identify the latest version of the EXEC:
  - If you just regenerated the EXEC in the previous step, you already have the latest level.
  - If you did not regenerate the EXEC, use the associated AUX files to determine the latest PTF against the given EXEC. This PTF number is the top one in the AUX file. Locate the PTF-numbered version of the EXEC (with filetype EXC*nnnnn*, where *nnnnn* is the PTF number). This PTF-numbered version of the EXEC is the version you want to use.
  - If the EXEC is not maintained with updates, it will not have any associated AUX files and you have to use the internal documentation in the EXEC to determine the PTF-numbered version to use. **Do not depend on date and time stamps**; examine the actual files. Each file ends with an update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn*, and XED*nnnnn* are exact images of the corresponding EXEC and XEDIT executables.
- Copy the PTF-numbered version that you just selected to its executable filetype on the alternate DELTA disk. This step does not apply to those EXECs that you regenerated with the EXECUPDT command.

  `copyfile / = EXEC` *fm* `(olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. *fm* is the alternate DELTA disk (593).

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Copy the new executable version of the module to the appropriate build disks:
  - If an executable version (with filetype EXEC) exists on the BUILD1 disk, copy the new version there.
  - If an executable version exists on the TASK disk, copy the new version there.
  - If an executable version does not reside on one of these disks, refer to the documentation provided with the PTF to determine where it should reside.

  `copyfile / = =` *fm*`2 (olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. You are copying these files from the alternate DELTA disk (593) to the BUILD1 disk (490), the TASK disk (5E5), or both.

  Use filemode 2 for files copied to the BUILD1 disk so they can be seen by other users.

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Return to the original filelist screen, using **PF3**.
- When you have processed each file in the original filelist screen, use **PF3** to exit filelist.

4. Repeat substep 3 for XEDIT macros. Replace EXEC with XEDIT and EXC*nnnnn* with XED*nnnnn*.

Step 9. Create Test EXECs and Files

In this step, you will create the EXECs and files needed to establish test named saved systems and saved segments.

1. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cms ■
```

2. Choose temporary names for all your new named saved systems and saved segments so that you do not overlay those you are working with until you are satisfied that the new ones work. This **sample procedure** uses the following names:

Real Name	Temporary Name
CMS	ALTCMS
CMSXA	ALTCMSXA
DOSBAM	ALTDSBAM
DOSINST	ALTDOSIN
CMSDOS	ALTDOS
CMSBAM	ALTBAM
CMSVSAM	ALTVSAM
CMSAMS	ALTAMS
INSTHELP	ALTINSHP
CMSINST	ALTINST
HELP	ALTHELP
GCS	ALTGCS

3. Copy the SAMPNSS EXEC:

```
vmfsetup 56643089 cms ■
copy sampnss exec fm testnss = = ■
```

4. Edit the TESTNSS EXEC, changing all the real names to temporary names:

```
xedit testnss exec ■
:
```

When you have finished, the TESTNSS EXEC should look like this:

```
/*************************************************************/
/**     Virtual Machine / System Product       5664-308    **/
/**     Contains restricted materials of IBM               **/
/**     Copyright (c) I B M   Corporation          1988    **/
/**     Licensed Materials -  Property of I B M            **/
/**     Refer to Copyright Instructions: Form G120-2083    **/
/*************************************************************/
arg parm1 parm2
Address Command
Select
  when parm1 = 'ALTCMS' & parm2 = 'ALTCMSXA' then
    do;
      'CP DEFSYS ALTCMS   0-A EW 20-22  EW E00-FFF SR MINSIZE=256K'
      'CP DEFSYS ALTCMSXA 0-A EW 20-22  EW E00-FFF SR MINSIZE=256K'
    end;
```

```
  WHEN PARM1 = 'ALTGCS' THEN
    'CP DEFSYS ALTGCS  0-6 EW 400-5FF SW MINSIZE=256K VMGROUP RSTD'
  when parm1 = 'ALTINST' then
    'CP DEFSEG ALTINST C00-C4F SR SPACE ALTINSHP'
  WHEN PARM1 = 'ALTHELP' THEN
    'CP DEFSEG ALTHELP C50-C9F SR SPACE ALTINSHP'
  WHEN PARM1 = 'ALTDOS' THEN
    'CP DEFSEG ALTDOS B00-B0F SR SPACE ALTDSBAM'
  WHEN PARM1 = 'ALTBAM' THEN
    'CP DEFSEG ALTBAM B10-B3F SR SPACE ALTDSBAM'
  WHEN PARM1 = 'ALTVSAM' THEN
    'CP DEFSEG ALTVSAM BA0-BFF SR A30-A3F EW SPACE ALTDSBAM'
  WHEN PARM1 = 'ALTAMS' THEN
    'CP DEFSEG ALTAMS B40-B9F SR A00-A2F EW SPACE ALTDSBAM'
  Otherwise
    do;
      say 'Unrecognized Parameters passed - ' parm1 parm2
    end;
End  /* Select */
'CP QUERY NSS ALL'
Exit
```

5. File the TESTNSS EXEC:

```
file ■
```

6. Copy the latest version of the SAMGEN EXEC:

```
copy samgen exec fm altsamgn = = ■
```

7. Edit the ALTSAMGN EXEC, changing CMSBAM to ALTBAM throughout:

```
xedit altsamgn exec ■
change/CMSBAM/ALTBAM/* * ■
file ■
```

8. Copy CMSBAM MAP and CMSBAM DOSLIB as ALTBAM MAP and ALTBAM DOSLIB:

```
copy cmsbam map fm altbam = = ■
copy cmsbam doslib fm altbam = = ■
```

9. Copy the latest version of the VSAMGEN EXEC:

```
copy vsamgen exec fm altvsamg = = ■
```

10. Edit the ALTVSAMG EXEC, changing CMSVSAM to ALTVSAM, and CMSAMS to ALTAMS throughout:

```
xedit altvsamg exec ■
change/CMSVSAM/ALTVSAM/* * ■
top ■
change/CMSAMS/ALTAMS/* * ■
file ■
```

11. Copy all files with a filename of CMSVSAM, CMSAMS, or CMSAMS*x* giving the copies filetypes of ALTVSAM, ALTAMS, and ALTAMS*x*:

```
filelist cmsvsam * * ■
copy cmsvsam ft fm altvsam = = ■
filelist cmsams* * * ■
copy cmsams ft fm altams = = ■
copy cmsamsx ft fm altamsx = = ■
```

12. You may have created a load list for the DCSSGEN command during the installation process. The one created in the sample installation process was called INSTLIST FILE. It should be on your A-disk (191). If it is not there, you probably used the IBM-supplied load list. This load list is called CMSINST EXECLIST. It should be on your 193 disk. Copy one of these load lists as ALTINST EXECLIST:

```
copy fn ft fm1 altinst execlist fm2 ■
```

*fm*2 is the 193 disk.

Step 10. Build Test Named Saved Systems

1. Use the TESTNSS EXEC to define segments and save your new CMS and CMSXA:

```
testnss altcms altcmsxa ■
```

The TESTNSS EXEC issues the DEFSYS command.

```
HCPNSD440I The Named Saved System (NSS) ALTCMS
            was successfully defined in fileid
            fileno.
HCPNSD440I The Named Saved System (NSS) ALTCMSXA
            was successfully defined in fileid
            fileno.
```

Note: The names are positional on the TESTNSS command (the System/370 name followed by the 370-XA name).

The TESTNSS EXEC first issues the DEFSYS command to define a skeleton system data file for ALTCMS and for ALTCMSXA. These messages tell you the DEFSYS commands were successful.

```
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTCMSXA NSS      MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The TESTNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

2. Issue the QUERY NSS ALL MAP command to make sure ALTCMS is defined properly. Verify that the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn ALTCMS   NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn ALTCMSXA NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
⋮
```

3. Set your machine mode to 370, load CMS (IPL 490), and save the ALTCMS system:

```
set machine 370 ■
```

```
ipl 490 clear parm savesys altcms ■
```

Load 490 with the option to save the system under the name ALTCMS.

```
HCPNSS440I The named saved system ALTCMS was
            successfully saved in fileid fileno.
VM/XA ALTCMS mm/dd/yy
```

```
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Press ENTER to initialize ALTCMS.

4. Set your machine mode to XA, load CMS (IPL 490), and save the ALTCMSXA system:

```
set machine xa ■
ipl 490 clear parm savesys altcmsxa ■
```

Load 490 with the option to save the system under the name ALTCMSXA.

```
HCPNSS440I The named saved system ALTCMSXA was
            successfully saved in fileid fileno.
VM/XA ALTCMS mm/dd/yy
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize ALTCMSXA.

5. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the ALTCMS (or ALTCMSXA) named saved system. The example below shows loading ALTCMS in System/370 mode:

```
set machine 370 ■
define storage 2m ■
STORAGE =      2M
STORAGE CLEARED - SYSTEM RESET
```

Defining storage causes a system reset.

```
ipl altcms ■
VM/XA ALTCMS mm/dd/yy
■
Ready; T=n.nn/n.nn hh:mm:ss
```

After receiving the version identification, press **ENTER** to complete ALTCMS or ALTCMSXA initialization.

6. Check the ALTCMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Verify that the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, verify that the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMSXA NSS      MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn ALTCMS   NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                              00020  00022   EW
                              00E00  00FFF   SR
nnnn ALTCMSXA NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                              00020  00022   EW
                              00E00  00FFF   SR
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

From this point until you do Step 17, IPL your alternate (test) CMS system (ALTCMS or ALTCMSXA) instead of your current (production) CMS.

Step 11. Install Test CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments

Warning: Do not **skip this step even if you have not changed CMSDOS, CMSBAM, CMSVSAM, or CMSAMS.**

The test CMSDOS and CMSBAM segments must **be installed before you install the test CMSVSAM and CMSAMS segments.**

1. Define your storage as 16 megabytes and IPL your new CMS system:

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
ipl altcms ■
DMSINS327I The installation saved segment could not be loaded
```

```
VM/XA ALTCMS mm/dd/yy
```

This is a sample version header.

```
** DO NOT press ENTER! **
```

```
access (noprof ■
```

This command suppresses execution of MAINT's PROFILE EXEC.

2. Establish the appropriate minidisk access order:

```
access 5E5 b ■
access 193 e ■
```

3. Issue:

```
set emsg on ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You want to see any and all error messages during execution of the installation EXEC.

4. Define a segment into which the DOSGEN EXEC can load ALTDOS:

```
defseg altdosin 900-90f sr ■
```

```
HCPNSD440I  Saved segment ALTDOSIN was
              successfully defined in fileid
              fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

This command defines a 1MB segment for ALTDOSIN starting at 900000. You may place this segment anywhere below the segment spaces defined for ALTDOS, ALTBAM, ALTVSAM, and ALTAMS.

5. Invoke the DOSGEN EXEC with a load address and the name ALTDOSIN. The load address used for ALTDOSIN in this example is 900000.

```
dosgen 900000 altdosin ■
HCPNSS440I Saved segment ALTDOSIN was
            successfully saved in fileid
            fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWGN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the ALTDOSIN segment. Error messages for the DOSGEN EXEC are listed on page 459.

6. Re-IPL your new CMS system:

```
ipl altcms ■
DMSINS327I The installation saved segment could not be loaded
VM/XA ALTCMS mm/dd/yy
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

7. Define ALTDOS, ALTBAM, ALTAMS, and ALTVSAM:

```
set sysname cmsdos altdosin ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Use **cmsdos** in this command, not **altdos**.

```
testnss altdos ■
HCPNSD440I Saved segment ALTDOS was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMSXA NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTINST  DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTDSBAM DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTDOS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
testnss altbam ■
HCPNSD440I Saved segment ALTBAM was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss AMTBAM   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
testnss altams ■
HCPNSD440I Saved segment ALTAMS was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTAMS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
testnss altvsam ■
HCPNSD440I Saved segment ALTVSAM was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss ALTVSAM  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Invoke the DOSGEN EXEC with a load address and the name ALTDOS. The load address set up by TESTNSS is B00000 for the name ALTDOS.

```
access 193 e ■
dosgen b00000 altdos ■
```

The load address and name are those recommended for the ALTDOS segment. Error messages for the DOSGEN EXEC are listed on page 459.

```
HCPNSS440I Saved segment ALTDOS was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSGEN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

9. To save the load map, rename it and copy it to the CMS BASE1 minidisk (193):

```
copy load map a altdos segmap e (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Error Messages from DOSGEN

If DOSGEN detects an error in the address that you specified:

```
DMSGEN095E INVALID ADDRESS
```

If DOSGEN cannot find a read/write accessed A-disk:

```
DMSGEN006E NO READ/WRITE A-DISK ACCESSED
```

If DOSGEN finds unresolved external references while loading the text files:

```
DMSGEN111E DOSGEN FAILED DUE TO LOAD ERRORS
```

If DOSGEN detects an error while assigning the storage key or saving the segment:

```
DMSGEN412S DOSGEN FAILED DUE TO SETKEY ERRORS

DMSGEN141S DOSGEN FAILED DUE TO SAVESYS ERRORS
```

10. Replace the name CMSBAM (not ALTBAM) in the CMS SYSNAME table with any name that is **not** previously used as a segment name:

```
set sysname cmsbam sysname ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The SET SYSNAME command enters an alternate name for CMSBAM in the SYSNAME table. Choose a name that was **not** used previously as a segment name. This command does not actually change the saved segment's name, but prevents CP from finding and IPLing the segment at the wrong time.

11. Place your CMS virtual machine in a CMS/DOS environment, then invoke ALTSAMGN to load the ALTBAM segment. You must give the EXEC an address at which to load the ALTBAM segment; this address also must match the address in the skeleton ALTBAM segment that you defined beforehand.

```
set dos on ■
```

This command places your virtual machine in a CMS/DOS environment.

```
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
             at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
altsamgn ■
DMSSGN363R ENTER LOCATION WHERE ALTBAM
WILL BE LOADED AND SAVED:
b10000 ■
DMSSGN364I FETCHING ALTBAM...
DMSFET710I PHASE DMSVBM ENTRY POINT AT LOCATION B100C0.
DMSSGN366R ENTER NAME OF SYSTEM TO BE SAVED:
altbam ■
HCPNSS440I Saved segment ALTBAM was successfully
            saved in fileid fileno
DMSSGN365I SYSTEM ALTBAM SAVED
Ready; T=n.nn/n.nn hh:mm:ss
```

sysname is the name you chose in substep 10.

The messages indicate that the segment has been loaded and saved.

12. Access the disk that you defined for VSAM when you were installing the system (see Step 28 in Chapter 2, Step 28 in Chapter 3, or Step 30 in Chapter 4) as your A-disk.

```
access vdevno a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

This puts the minidisk (*vdevno*) in read/write mode.

13. Invoke ALTVSAMG EXEC:

```
altvsamg ■
SELECT ONE OF THE FOLLOWING FUNCTIONS BY ENTERING THE NUMBER:

 1. INSTALL AMS           (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 2. INSTALL VSAM          (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 3. INSTALL VSAM AND AMS  (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 4. BUILD  AMS            (BUILD DOSLIB, CREATE SEGMENT)
 5. BUILD  VSAM           (BUILD DOSLIB, CREATE SEGMENT)
 6. BUILD  VSAM AND AMS   (BUILD DOSLIB, CREATE SEGMENT)
 7. RESTART AMS           (CREATE SEGMENT)
 8. RESTART VSAM          (CREATE SEGMENT)
 9. RESTART VSAM AND AMS  (CREATE SEGMENT)
10. QUIT                  (EXIT ALTVSAMG EXECUTION)

      ENTER RESPONSE...
```

```
6 ■
```

Choosing option 6 tells the EXEC to create both ALTVSAM and ALTAMS segments as new segments, without reading the text files from the VSAM product tape.

If you want to install a new release of VSAM, instead of servicing the current release, you must erase all files associated with VSAM before you issue ALTVSAMG; then, when you issue ALTVSAMG, you must choose option 3.

```
DMSVGN365R ONE OR MORE OF THE TEXT FILES LISTED IN THE CMSVSAM EXEC
            ARE MISSING.  THE VSAM PP PID TAPE SHOULD BE ON TAPE DRIVE 181
            TO RESTORE THE FILES. ENTER:
            'GO' IF TAPE DRIVE IS READY TO LOAD FILES,
            'QUIT' TO STOP GENERATION PROCESS.
go ■
```

Messages

While ALTVSAMG is processing, you may receive error and information messages. These messages are self-explanatory. Messages labeled 2101I are information messages from the linkage editor and may be ignored.

Respond to the EXEC messages as they appear:

```
DMSVGN362I LINK-EDITING ALTVSAM ...
DMSVGN363I ALTVSAM DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
            OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE ALTVSAM SHARED WILL BE
         LOADED AND SAVED:

ba0000 ■

DMSVGN363R ENTER LOCATION WHERE ALTVSAM NONSHARED WILL BE
         LOADED AND SAVED:
a30000 ■
```

```
DMSVGN364I FETCHING ALTVSAM ...
DMSFET710I PHASE DMSVVS ENTRY POINT AT LOCATION nnnnnn
DSMVGN371R ALTVSAM IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
altvsam ■
```

This name is the default name with which the segment was defined.

```
DMSVGN365I SYSTEM ALTVSAM SAVED.

DMSVGN368R ERASE ALTVSAM DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
DMSVGN362I LINK-EDITING ALTAMS ...
DMSVGN363I ALTAMS DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
             OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE ALTAMS SHARED WILL BE
          LOADED AND SAVED:
b40000 ■
```

This is the recommended location.

```
DMSVGN363R ENTER LOCATION WHERE ALTAMS NONSHARED WILL BE
          LOADED AND SAVED:
a00000 ■
DMSVGN364I FETCHING ALTAMS ...
DMSFET710I PHASE DMSVAS ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAN ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAX ENTRY POINT AT LOCATION nnnnnn
DMSVGN371R ALTAMS IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
altams ■
```

This name is the default name used for defining the segment.

```
DMSVGN365I SYSTEM ALTAMS SAVED.

DMSVGN368R ERASE ALTAMS DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
Ready; T=n.nn/n.nn hh:mm:ss
```

14. Set DOS off:

```
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: If you loaded files from the VSAM product tape in substep 13 on page 461, do not erase them. You will need them in Step 18.

Step 12. Install Test CMSINST and HELP Saved Segments

Warning: Do not **skip this step even if you have not changed CMSINST or HELP.**

1. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cp ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Make sure that you have the right NAMESAVE segments in your user directory:
   - You must have the NAMESAVE ALTHELP statement in your user directory in order to save the HELP file directory information in a test saved segment.
   - If the ALTHELP segment is defined as a member of a segment space, you must have a NAMESAVE statement (for example, NAMESAVE ALTINSHP) in the directory entry for the user who invokes SAVEFD. SAVEFD specifies the name of the segment space.

   If you have to change the directory, be sure to issue DIRECTXA to bring the changed directory online.
3. Log on as MAINT (unless you are continuing from the previous step).
4. Check your virtual storage. If it is less than 16MB, issue the following:

```
define storage 16m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 0016M
```

5. IPL your new CMS system:

```
ipl altcms ■
DMSWSP327I The installation saved segment could not be loaded
```

```
VM/XA ALTCMS mm/dd/yy
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

This is a sample version header. If you defined your own version heading, your own heading will appear.

Press **ENTER** to complete the CMS initialization.

6. Release the INSTHELP segment so that you can define a test segment in the same space:

```
segment release insthelp ■
```

7. Define a segment for ALTINST:

```
testnss altinst ■
HCPNSD440I Saved segment ALTINST was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

8. Define the ALTHELP segment:

```
testnss althelp ■
HCPNSD440I Saved segment ALTHELP was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTHELP  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Save both segments:

```
saveseg altinst ■
```

Each member saved segment defined by SAMPNSS must be saved separately.

```
HCPNSS440I Saved segment ALTINST was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg althelp ■
HCPNSS440I Saved segment ALTHELP was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Redefine each segment to create the skeleton segments:

```
testnss altinst ■
HCPNSD440I Saved segment ALTINST was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

testnss althelp ■
HCPNSD440I Saved segment ALTHELP was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm ALTHELP  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

11. Using the load list you copied in Step 9, issue the DCSSGEN command as follows:

```
access 193 e ■
dcssgen altinst execlist e altinst ■
```

ALTINST EXECLIST E is the file ID of the file that contains the list of EXECs and editor macros to be loaded into the test segment.

ALTINST is the name of the test segment.

```
HCPNSS440I Saved segment ALTINST was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: When you built your ALTCMS nucleus, if you indicated in the DMSNGP file (USEINST = YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs ALTCMS. If you previously indicated that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild ALTCMS.

Messages from DCSSGEN Command

While DCSSGEN is processing, you may receive error or warning messages that indicate specific conditions. If errors were encountered, after processing is complete you receive the following prompt:

```
DMSEXG298R An error has been detected while building the
           DCSS.  Do you still want the DCSS saved?
           Enter 1 (YES) or 0 (NO).
```

Enter **1** to disregard the error(s) and save the segment, or enter **0** to not save the segment. If you do not save the segment, you receive the message:

DMSEXG288I *segname* not saved

If DCSSGEN encounters an error while saving the segment, you receive the message:

DMSEXG288E *dcssname* not saved

If your virtual machine is not large enough to contain the segment (you need 16M), you receive the message:

DMSEXG284E The DCSS is not completely inside the virtual machine

To correct this situation, increase the size of your virtual machine, re-IPL ALTCMS, and reissue the DCSSGEN command.

12. Define your virtual storage **less** than the address at which the ALTHELP segment is to be loaded. For example, if the ALTHELP segment is defined at X'C50000', define your storage as 12MB.

```
define storage 12m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 12M
```

13. Re-IPL your test CMS system:

```
ipl altcms ■
VM/XA ALTCMS mm/dd/yy                    This is a sample version header.
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

14. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cms ■
```

15. Issue the following commands to initialize and save the segment:

    (For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*. For more information on saved segments, refer to *VM/XA SP Guide to Saved Segments*.)

```
savefd init vaddr label althelp ■
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is 49D for mixed-case American English, 49C for uppercase American English.

label is the CMS label assigned to the disk. (In this sample procedure, it is MNT49D or MNT49C.)

```
segment purge altinst ■
Ready; T=n.nn/n.nn hh:mm:ss
savefd save vaddr label althelp ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment ALTHELP was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

16. Verify that the ALTINST and ALTHELP segments were defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss ALTINSHP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss ALTINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss ALTHELP     NSS      MAINT
```

17. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl altcms ■
VM/XA ALTCMS mm/dd/yy
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 13. Test the System

Now that you have applied service to all necessary files, you must test the system to make sure that the problem has been fixed.

1. Test the system.
2. To allow selected users to test the system, have them issue the following commands:

```
#cp link maint 490 490 rr ■
password ■
set sysname cmsdos altdos ■
set sysname cmsbam altbam ■
set sysname cmsvsam altvsam ■
set sysname cmsams altams ■
ipl altcms ■
```

If the service performs to your satisfaction, perform Steps 14 – 19 to rebuild your system on the 190 disk. If the service fails, then you must find the problem, correct it, and re-test the system.

Step 14. Purge the Test Named Saved Systems and Saved Segments

1. Issue QUERY NSS ALL to determine the spool IDs of the test named saved systems and saved segments segments you created in Step 10, Step 11, and Step 12 (ALTCMS, ALTCMSXA, ALTDSBAM, ALTDOSIN, ALTDOS, ALTBAM, ALTVSAM, ALTAMS, ALTINSHP, ALTINST, and ALTHELP):

```
query nss all ■
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMS   NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTCMSXA NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTDSBAM DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTDOSIN DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTDOS   DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTBAM   DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTVSAM  DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTAMS   DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTINSHP DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTINST  DCSS     MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss ALTHELP  DCSS     MAINT
```

The spool ID is the number in the second column, under the heading **FILE**.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Purge the named saved systems and saved segments:

```
purge nss spoolid1 ... spoolidn ■
```

You can list as many spool IDs as necessary on a single PURGE NSS command.

The PURGE NSS command will take effect as soon as all users stop using the named saved systems and saved segments.

Step 15. Put the New CMS System into Production

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cms ■
   ```

2. Rename DMSNGP TEXT to save it. Then edit the DMSNGP ASSEMBLE file and make the following changes:
   a. Change SYSDISK=490 to SYSDISK=190.
   b. Change IPLADDR=490 to IPLADDR=190.
   c. Change HELP=49D to HELP=19D (or HELP=49C to HELP=19C).
   d. Change SYSNAME=*newname* to SYSNAME=CMS, where *newname* is the name you choose for your test system. The sample procedure in this chapter uses ALTCMS.
   e. Change INSTSEG=*newname* to SYSNAME=CMSINST, where *newname* is the name you choose for your test installation segment. The sample procedure in this chapter uses ALTINST.
   f. Change the VERSION= and INSTID= parameters to identify your new CMS system.

   These changes will cause the new CMS nucleus (with service) to be rebuilt on the 190 minidisk.

3. Reassemble the DMSNGP ASSEMBLE file:

   ```
   vmfhasm dmsngp 56643089 cms ■
   ```

4. Copy DMSNGP TEXT to the CMS alternate LOCAL1 disk (395):

   ```
   copy dmsngp text a = = fm (replace ■
   ```

   fm is the CMS alternate LOCAL1 disk (395).

5. Use the DASD DUMP Restore program to copy the 490 disk to the 190 disk:

   Note: Access the 190 disk as R/W, using an unassigned filemode.

   ```
   ddr ■
   VM/XA SYSTEM PRODUCT DASD DUMP RESTORE PROGRAM
   ENTER CARD READER ADDRESS OR CONTROL STATEMENTS.
   ENTER:
   ```

   ```
   sysprint cons ■
   ENTER:
   ```

   This command tells DDR to send program messages to your console.

   ```
   input 490 devtype MNT490 ■
   ENTER:
   ```

   490 is your test CMS system disk. *devtype* is the device type of the DASD volume where 490 is located.

   ```
   output 190 devtype MNT190 ■
   ENTER:
   ```

   190 is your new CMS system disk. *devtype* is the device type of the DASD volume where 190 is located.

```
copy 000 endcyl ■
```

The value for *endcyl* depends on the device type of your 490 disk:

Device Type	*endcyl*
3350	73
3375	112
3380	71
3390	67

```
DMKDDR711R Volid Read is MNT490.  Do you
             wish to continue?
yes ■
ENTER NEXT EXTENT OR NULL LINE:
■
⋮
END OF COPY
ENTER:
■
END OF JOB
```

You have not yet changed the label of the 190 minidisk. (You do so in substep 6 on page 470.)

6. Re-label the 190 minidisk:

```
access 190 c ■
DMSACP723I C(190) R/O
DMSACC725I 190 ALSO = S DISK
```

```
format 190 c (label ■
ENTER LABEL:
mnt190 ■
Ready; T=n.nn/n.nn hh:mm:ss
release c ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You **must** use the **label** option. If you do not, you erase all the files on the 190 minidisk.

7. Repeat substeps 5 on page 469 and 6 to copy the alternate HELP disks (49D, 49C, 49B...) to their corresponding current disks (19D, etc.) and to re-label the current disks as MNT19*n*. Check the minidisk definitions in your user directory to find the appropriate value for *endcyl*.
8. Erase DMSNGP TEXT on your A-disk:

```
erase dmsngp text a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 16. Rebuild the Nucleus

1. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

2. Generate the new CMS nucleus:

```
vmfbld 56643089 cms (punch ■
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
⋮
```

This invokes the VMFBLD EXEC to build a new nucleus for CMS. The VMFBLD EXEC performs a number of system generation functions for you. For more information on this EXEC, see "VMFBLD EXEC" on page 652.

```
RDR FILE fileno SENT FROM MAINT PUN AS fileno
   RECS nnnn COPY 001 N NOHOLD NOKEEP
```

The system loader punches the CMS nucleus. This message tells you that the punch file is complete.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF which is now being applied has another PTF, outside CMS, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

3. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

4. Ensure that the proper files are in your virtual reader:

```
query rdr * all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

6. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

7. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

8. IPL your virtual reader:

```
ipl 00c ■

DMSINS327I  The installation saved segment could not be loaded
```

Informational message: The CMSINST installation segment has not been loaded and saved yet.

```
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM

Ready; T=n.nn/n.nn hh:mm:ss
```

The ready message indicates that the nucleus has been loaded to your 190 minidisk successfully.

9. Receive the CMS load map into your reader:

```
spool prt * nohold ■
close prt * ■
```

10. Query the reader to identify the CMS load map:

```
query rdr * all ■

ORIGINID FILE   CLASS RECORDS  CPY HOLD DATE  TIME     NAME     TYPE DIST
MAINT    fileno M PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss              SYSPROG
```

The CMS load map is the file with a blank filename and filetype. It has approximately 8,100 records. The exact size varies according to the system default language, local modifications, and VMSUP level. Note the *fileno* of this file. You will use it in your next command.

```
vmfsetup 56643089 cms ■
receive fileno fn ft fm ■
fn ft fm created
DMSRDC738I Record length is 132 bytes
File fn ft D received from MAINT at
  nodeid as fn ft fm
Ready; T=n.nn/n.nn hh:mm:ss
```

The CMS load map is loaded on the CMS LOCAL1 (395) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1–9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Note: From this point on, to IPL your new production system, issue `ipl 190` ■.

Step 17. Rebuild Named Saved Systems

1. Use the SAMPNSS EXEC to define segments and save your new CMS and CMSXA:

```
sampnss cms cmsxa ■

HCPNSD440I The Named Saved System (NSS) CMS was
            successfully defined in fileid
            fileno.
HCPNSD440I The Named Saved System (NSS) CMSXA was
            successfully defined in fileid
            fileno.
```

The SAMPNSS EXEC issues the DEFSYS command.

Note: The names are positional on the SAMPNSS command (the System/370 name followed by the 370-XA name).

The SAMPNSS EXEC first issues the DEFSYS command to define a skeleton system data file for CMS and for CMSXA. These messages tell you the DEFSYS commands were successful.

```
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

The SAMPNSS EXEC issues a QUERY NSS command. These messages show what system data files are defined.

2. Issue the QUERY NSS ALL MAP command to make sure CMS is defined properly. Verify that the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown.

```
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  S  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
⋮
```

3. Set your machine mode to 370, load CMS (IPL 190), and save the CMS system:

```
set machine 370 ■

ipl 190 clear parm savesys cms ■

HCPNSS440I The named saved system CMS was
            successfully saved in fileid fileno.
VM/XA CMS 5.6 mm/dd/yy hh:mm

■

Ready; T=n.nn/n.nn hh:mm:ss
```

Load 190 with the option to save the system under the name CMS.

Press **ENTER** to initialize CMS:

4. Set your machine mode to XA, load CMS (IPL 190), and save the CMSXA system:

```
set machine xa ■
ipl 190 clear parm savesys cmsxa ■
```

Load 190 with the option to save the system under the name CMSXA.

```
HCPNSS440I The named saved system CMSXA was
           successfully saved in fileid fileno.
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Press **ENTER** to initialize CMSXA.

5. Set your machine mode to 370 (or XA), redefine your virtual storage to 2 megabytes, and load the CMS (or CMSXA) named saved system. The example below shows loading CMS in System/370 mode:

```
set machine 370 ■
define storage 2m ■
STORAGE =      2M
STORAGE CLEARED - SYSTEM RESET
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

Defining storage causes a system reset.

After receiving the version identification, press **ENTER** to complete CMS or CMSXA initialization.

6. Check the CMS status by issuing QUERY NSS ALL and QUERY NSS ALL MAP. Verify that the information under FILENAME and FILETYPE in the response to QUERY NSS ALL is the same as shown. Also, verify that the information under BEGPAG, ENDPAG, TYPE, and CL in the response to QUERY NSS ALL MAP is the same as shown. The system data file should now have class A (rather than S) and have one user (MAINT).

```
query nss all ■
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
query nss all map ■
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS VMGROUP
⋮
nnnn CMS      NSS      000256K  00000  0000A   EW  A  00001  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
nnnn CMSXA    NSS      000256K  00000  0000A   EW  A  00000  OMITTED  NO
                                00020  00022   EW
                                00E00  00FFF   SR
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 18. Reinstall the CMSDOS, CMSBAM, CMSVSAM, and CMSAMS Saved Segments

Warning: Do not skip this step even if you have not changed CMSDOS, CMSBAM, CMSVSAM, or CMSAMS.

The CMSDOS and CMSBAM segments must be installed before you install the CMSVSAM and CMSAMS segments.

1. Define your storage as 16 megabytes and IPL your new CMS system:

```
define storage 16m ■
STORAGE = 0016M
STORAGE CLEARED - SYSTEM RESET
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
** DO NOT press ENTER! **
access (noprof ■
```

This command suppresses execution of MAINT's PROFILE EXEC.

2. Establish the appropriate minidisk access order:

```
access 5E5 b ■
access 193 e ■
```

3. Issue:

```
set emsg on ■
Ready; T=n.nn/n.nn hh:mm:ss
```

You want to see any and all error messages during execution of the installation EXEC.

4. Define a segment into which the DOSGEN EXEC can load CMSDOS:

```
defseg dosinst 900-90f sr ■
HCPNSD440I  Saved segment DOSINST was
              successfully defined in fileid
              fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

This command defines a 1MB segment for DOSINST starting at 900000. You may place this segment anywhere below the segment spaces defined for CMSDOS, CMSBAM, CMSVSAM, and CMSAMS.

5. Invoke the DOSGEN EXEC with a load address and the name DOSINST. The load address used for DOSINST in this example is 900000.

```
dosgen 900000 dosinst ■
```

```
HCPNSS440I Saved segment DOSINST was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSWGN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

The load address and name are those recommended for the DOSINST segment. Error messages for the DOSGEN EXEC are listed on page 459.

6. Re-IPL your new CMS system:

```
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

7. Establish the appropriate minidisk access order:

```
access 193 e ■
```

8. Define CMSDOS, CMSBAM, CMSAMS, and CMSVSAM:

```
set sysname cmsdos dosinst ■
Ready; T=n.nn/n.nn hh:mm:ss
sampnss cmsdos ■
HCPNSD440I Saved segment CMSDOS was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA    NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss GCS      NSS      MAINT
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss DOSBAM   DCSS     MAINT
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSDOS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsbam ■
HCPNSD440I Saved segment CMSBAM was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSBAM   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss

sampnss cmsams ■
HCPNSD440I Saved segment CMSAMS was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
:
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSAMS   DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

```
sampnss cmsvsam ■
HCPNSD440I Saved segment CMSVSAM was successfully
            defined in fileid fileno
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
⋮
*NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss CMSVSAM  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Invoke the DOSGEN EXEC with a load address and the name CMSDOS. The load address set up by SAMPNSS is B00000 for the name CMSDOS.

```
dosgen b00000 cmsdos ■
```

The load address and name are those recommended for the CMSDOS segment. Error messages for the DOSGEN EXEC are listed on page 459.

```
HCPNSS440I Saved segment CMSDOS was successfully
            saved in fileid fileno
PRT FILE fileno SENT FROM MAINT PRT AS fileno
  RECS nnnn COPY 001 A NOHOLD NOKEEP
DMSGEN715I DOSGEN COMPLETE
Ready; T=n.nn/n.nn hh:mm:ss
```

10. To save the load map, rename it and copy it to the 193 minidisk (E):

```
copy load map a cmsdos segmap e (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

11. Replace the name CMSBAM in the CMS SYSNAME table with any name that is **not** previously used as a segment name:

```
set sysname cmsbam sysname ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The SET SYSNAME command enters an alternate name for CMSBAM in the SYSNAME table. Choose a name that was **not** previously used as a segment name. This command does not actually change the saved segment's name, but prevents CP from finding and IPLing the segment at the wrong time.

12. Place your CMS virtual machine in a CMS/DOS environment, then invoke SAMGEN to load the CMSBAM segment. You must give the EXEC an address at which to load the CMSBAM segment; this address also must match the address in the skeleton CMSBAM segment that you defined beforehand.

```
set dos on ■
```

This command places your virtual machine in a CMS/DOS environment.

```
DMSSET400I SYSTEM sysname DOES NOT EXIST
DMSSET1101I 100K DOS partition defined
             at hexadecimal location 020000.
Ready; T=n.nn/n.nn hh:mm:ss
samgen ■
DMSSGN363R ENTER LOCATION WHERE CMSBAM
WILL BE LOADED AND SAVED:
```

sysname is the name you chose in substep 11.

```
b10000 ■
DMSSGN364I FETCHING CMSBAM...
DMSFET710I PHASE DMSVBM ENTRY POINT AT LOCATION B100C0.
DMSSGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsbam ■

HCPNSS440I Saved segment CMSBAM was successfully
           saved in fileid fileno
DMSSGN365I SYSTEM CMSBAM SAVED
Ready; T=n.nn/n.nn hh:mm:ss
```

The messages indicate that the segment has been loaded and saved.

13. Access the VSAM product disk (the same disk you accessed in Step 11, substep 12 on page 460) as your A-minidisk.

```
access vdevno a ■
Ready; T=n.nn/n.nn hh:mm:ss
```

14. Invoke VSAMGEN EXEC:

```
vsamgen ■
SELECT ONE OF THE FOLLOWING FUNCTIONS BY ENTERING THE NUMBER:

 1. INSTALL AMS           (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 2. INSTALL VSAM          (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 3. INSTALL VSAM AND AMS  (READ VSAM PRODUCT TAPE, BUILD DOSLIB, CREATE SEGMENT)
 4. BUILD  AMS            (BUILD DOSLIB, CREATE SEGMENT)
 5. BUILD  VSAM           (BUILD DOSLIB, CREATE SEGMENT)
 6. BUILD  VSAM AND AMS   (BUILD DOSLIB, CREATE SEGMENT)
 7. RESTART AMS           (CREATE SEGMENT)
 8. RESTART VSAM          (CREATE SEGMENT)
 9. RESTART VSAM AND AMS  (CREATE SEGMENT)
10. QUIT                  (EXIT VSAMGEN EXECUTION)

      ENTER RESPONSE...

6 ■
```

Choosing option 6 tells the EXEC to create both CMSVSAM and CMSAMS segments as new segments, without reading the text files from the VSAM product tape. If you read in new files from the VSAM product tape in substep 13 on page 461 of Step 11, you will use those files now.

Messages

While VSAMGEN is processing, you may receive error and information messages. These messages are self-explanatory. Messages labeled 2101I are information messages from the linkage editor and may be ignored.

Respond to the EXEC messages as they appear:

```
DMSVGN362I LINK-EDITING CMSVSAM ...
DMSVGN363I CMSVSAM DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
             OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSVSAM SHARED WILL BE
          LOADED AND SAVED:
ba0000 ■
DMSVGN363R ENTER LOCATION WHERE CMSVSAM NONSHARED WILL BE
          LOADED AND SAVED:
a30000 ■
DMSVGN364I FETCHING CMSVSAM ...
DMSFET710I PHASE DMSVVS ENTRY POINT AT LOCATION nnnnnn
DSMVGN371R CMSVSAM IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsvsam ■
```

This name is the default name with which the segment was defined.

```
DMSVGN365I SYSTEM CMSVSAM SAVED.

DMSVGN368R ERASE CMSVSAM DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
DMSVGN362I LINK-EDITING CMSAMS ...
DMSVGN363I CMSAMS DOSLIB CREATED ON DISK 'A'.
DMSVGN370R ENTER 'GO' IF SAVED SYSTEM IS TO BE CREATED,
             OTHERWISE ENTER 'QUIT'.
go ■
DMSVGN363R ENTER LOCATION WHERE CMSAMS SHARED WILL BE
          LOADED AND SAVED:
b40000 ■
```

This is the recommended location.

```
DMSVGN363R ENTER LOCATION WHERE CMSAMS NONSHARED WILL BE
          LOADED AND SAVED:
a00000 ■
DMSVGN364I FETCHING CMSAMS ...
DMSFET710I PHASE DMSVAS ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAN ENTRY POINT AT LOCATION nnnnnn
DMSFET710I PHASE DMSVAX ENTRY POINT AT LOCATION nnnnnn
DMSVGN371R CMSAMS IS LOADED, IF ZAPS ARE TO BE APPLIED GO INTO
        'CP' MODE, APPLY THE ZAPS AND THEN REPLY 'GO'
go ■
DMSVGN366R ENTER NAME OF SYSTEM TO BE SAVED:
cmsams ■
```

This name is the default name used for defining the segment.

```
DMSVGN365I SYSTEM CMSAMS SAVED.

DMSVGN368R ERASE CMSAMS DOSLIB ? ... ENTER 'YES' OR 'NO':
no ■
Ready; T=n.nn/n.nn hh:mm:ss
```

15. You may now purge the DOSINST named saved segment:

```
query nss all ■
OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME FILETYPE ORIGINID
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS      NSS      MAINT
⋮
*NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss DOSINST  DCSS     MAINT
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
purge nss fileno ■
No files purged
0001 file pending purged
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file identifier for DOSINST.

16. Set DOS off:

```
set dos off ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 19. Reinstall the CMSINST and HELP Saved Segments

> **Warning:** Do not **skip this step even if you have not changed CMSINST or HELP.**

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Make sure that you have the right NAMESAVE segments in your user directory:
   - You must have the NAMESAVE HELP statement in your user directory in order to save the HELP file directory information in a saved segment.
   - If the HELP segment is defined as a member of a segment space, you must have a NAMESAVE statement (for example, NAMESAVE INSTHELP) in the directory entry for the user who invokes SAVEFD. SAVEFD specifies the name of the segment space.
3. Log on as MAINT (unless you are continuing from the previous step).
4. Check your virtual storage. If it is less than 16MB, issue the following:

   ```
   define storage 16m ■
   STORAGE CLEARED - SYSTEM RESET
   STORAGE = 0016M
   ```

5. IPL your new CMS system:

   ```
   ipl cms ■
   VM/XA CMS 5.6 mm/dd/yy hh:mm
   ```

   ■ Press **ENTER** to complete the CMS initialization.

   ```
   SYNONYM SYN
   CP TERM MODE VM
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

6. Purge your old CMSINST segment:

   ```
   segment purge cmsinst ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

7. Define a segment for CMSINST:

   Note: If you have increased the number of entries in the CMS load list, be sure that the DEFSEG entry for CMSINST in the SAMPNSS EXEC has enough pages defined to accomodate the increased size.

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
           defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

   The EXEC issues a QUERY command that displays information about the segments defined for named saved systems.

8. Define the HELP segment:

```
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
           defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

9. Save both segments:

```
saveseg cmsinst ■
```

   Each member saved segment defined by SAMPNSS must be saved separately.

```
HCPNSS440I Saved segment CMSINST was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
saveseg help ■
HCPNSS440I Saved segment HELP was successfully
           saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Redefine each segment to create the skeleton segments:

```
sampnss cmsinst ■
HCPNSD440I Saved segment CMSINST was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
sampnss help ■
HCPNSD440I Saved segment HELP was successfully
            defined in fileid fileno
OWNERID FILE TYPE CL RECS DATE  TIME  FILENAME FILETYPE ORIGINID
*NSS    nnnn NSS  A  nnnn mm:dd hh:mm CMS      NSS      MAINT
⋮
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm CMSINST  DCSS     MAINT
*NSS    nnnn NSS  S  nnnn mm:dd hh:mm HELP     DCSS     MAINT
Ready; T=n.nn/n.nn hh:mm:ss
```

11. You may have created a load list for the DCSSGEN command during the installation process. The one created in the sample installation process was called INSTLIST FILE. It should be on your A-disk (191). If it is not there, you probably used the IBM-supplied load list. This load list is called CMSINST EXECLIST. It should be on your K-disk (193). Using one of these load lists, issue the DCSSGEN command as follows:

```
access 193 e ■
dcssgen fn ft fm cmsinst ■
```

fn ft fm is the fileid of the file that contains the list of EXECs and Editor macros to be loaded into the segment.

CMSINST is the name of the segment. CMSINST is the default name used for defining the segment and for specifying the segment name in DMSNGP or in response to the installation questions. If you do not specify a segment name, the default name is CMSINST.

```
HCPNSS440I Saved segment CMSINST was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

Note: When you built your CMS nucleus, if you indicated in the DMSNGP file (USEINST = YES) or in answer to the DMSINQ296R prompt that you wanted to use the installation segment (the default is YES), then this segment is used each time a user IPLs CMS. If you indicated earlier that you did not want to use the installation segment but now want to use it, you must modify the DMSNGP file to indicate that the segment should be used, then assemble the modified DMSNGP and rebuild CMS.

12. Define your virtual storage as **less** than the address at which the HELP segment is to be loaded. For example, if the HELP segment is defined at X'C50000', define your storage as 12MB.

```
define storage 12m ■
STORAGE CLEARED - SYSTEM RESET
STORAGE = 12M
```

13. Re-IPL your new CMS system:

```
ipl cms ■

VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

14. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cms ■
```

15. Issue the following commands to initialize and save the segment:

    (For more information about using the SAVEFD command to save minidisk file directory information in a saved segment, refer to *VM/XA SP CMS Command Reference*. For more information about saved segments, refer to *VM/XA SP Guide to Saved Segments*.)

```
savefd init vaddr label help ■
Ready; T=n.nn/n.nn hh:mm:ss
```

vaddr is 19D for mixed-case American English, 19C for uppercase American English.

label is the CMS label assigned to the disk. (The labels in the sample directory are MNT19D and MNT19C.)

```
savefd save vaddr label help ■
DMSACP723I Z(19D) R/O
HCPNSS440I Saved segment HELP was successfully
            saved in fileid fileno
Ready; T=n.nn/n.nn hh:mm:ss
```

16. Verify that the CMSINST and CMSHELP segments are defined correctly:

```
query nss all ■
OWNERID FILE   TYPE CL RECS DATE  TIME     FILENAME    FILETYPE ORIGINID
⋮
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss INSTHELP    NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss CMSINST     NSS      MAINT
*NSS    fileno NSS  A  nnnn mm/dd hh:mm:ss HELP        NSS      MAINT
```

17. Redefine your storage before you continue:

```
define storage 16m ■
STORAGE = 16M
Storage cleared - system reset
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
SYNONYM SYN
CP TERM MODE VM
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 20. Back Up the Named Saved Systems

Use the procedure in Step 30 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" to back up the new CMS and CMSXA named saved systems.

What to Do Next

- You have finished the service procedure for CMS.
- If you are also servicing CP, see Chapter 15, "Rebuilding CP after Applying Service" on page 487.

Chapter 15. Rebuilding CP after Applying Service

This chapter describes a step-by-step procedure for rebuilding CP after applying program update service, corrective service, or local service.

Should You Be Doing This Now?

- If you are also servicing CMS, you must rebuild CMS before you rebuild CP. If you have not yet rebuilt CMS, see Chapter 14, "Rebuilding CMS after Applying Service" on page 431.
- Otherwise, continue with this chapter.

Step 1. Build a New CP Macro Library

> **Note:** This step describes a method of updating the CP macro libraries that isolates all of the serviced macros and control blocks into a separate MACLIB using the VMFMAC EXEC. The new MACLIB is added to the MACS record in the HCPXA CNTRL file.
>
> An alternate method of updating the CP macro libraries is to update each affected MACLIB with the changed macro or control block by using the MACLIB command. (See *VM/XA SP CMS Command Reference* for details on the MACLIB command.)
>
> If you choose the alternate method of building new CP macro libraries, do so now and continue with "Step 2. Update the CP Message Repository" on page 491.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Determine the macros and control blocks you have service for:

   - Search the DELTA string for all macros and control blocks you have service for:

`listfile * A*HP fm1 (execs args fname ■`	*fm1* is the alternate DELTA (594) disk.
`listfile * A*HP fm2 (append args fname ■`	*fm2* is the intermediate alternate DELTA (591) disk.
`listfile * A*HP fm3 (append args fname ■`	*fm3* is the current DELTA (294) disk.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.
   - Save the filenames in the CPUPDATE EXEC and erase the CMS EXEC:

     ```
     copyfile cms exec a cpupdate = = (replace ■
     erase cms exec a ■
     ```

   - Determine which of the updated files are macros (filetype MACRO), and which are control blocks (filetype COPY). Store the results in the CPNEW EXEC:

     ```
     cpupdate listfile % macro * (append args ftype ■
     cpupdate listfile % copy * (append args ftype ■
     ```

     Ignore the messages that appear on your screen while these two commands are processing.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.

- Save the filenames in the CPNEW EXEC:

```
copyfile cms exec a cpnew = = (replace ■
```

- Edit the CPNEW EXEC to remove any duplicate entries:

```
xedit cpnew exec ■
sort * 1 15 ■
⋮
file ■
```

3. Determine the macros and control blocks for which you have local service. Follow your installation's naming conventions for locally serviced files. Locally serviced files should reside on the LOCAL1 disk string.
   - If no macros and control blocks have local service, continue with substep 4.
   - Otherwise, add your locally serviced macros and control blocks to CPNEW EXEC. When you are done, the CPNEW EXEC should list all macros and control blocks for which you have received IBM service (if any), plus any macros and control blocks for which you have local service.

     If you do not have a CPNEW EXEC on your A-disk, copy the HCPGPI EXEC to use as a template:

```
copyfile hcpgpi exec fm cpnew = a (recfm f lrecl 80 ■
```

     If you created the CPNEW EXEC by copying the HCPGPI EXEC, use the contents of the HCPGPI EXEC only for a guide to the format of the entries. After you have added the macro and control block names you need, in the proper format, delete the macro names that originated from the HCPGPI EXEC:

```
xedit cpnew exec a ■
⋮
file ■
```

4. If you have no updated macros or control files at this point, continue with "Step 2. Update the CP Message Repository" on page 491.
5. Generate the new macro library on your A-disk (191) by using the VMFMAC EXEC:

```
vmfmac cpnew hcpxa ■
DMSUPD178I Updating macroname MACRO fm
DMSUPD178I Applying macroname update fm
macroname MACRO ADDED
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

6. The contents of CPNEW COPY were appended to CPNEW MACLIB. You can now erase CPNEW COPY (a file created by VMFMAC) on your A-disk (191):

```
erase cpnew copy a ■
```

7. Add the new macro library name to the TEXT MACS card in the HCPXA CNTRL file.

   List the new macro library first so that you use the updated macros. If you have already done this during previous service to CP, you do not have to do this now.

```
xedit hcpxa cntrl fm ■
locate/TEXT MACS ■
TEXT MACS maclib ...
change/MACS/MACS CPNEW/ ■
TEXT MACS CPNEW maclib ...
file ■
```

   fm is the CP BASE1 disk (194).

8. Copy the changed files from your A-disk to the CP database: *fm* is the alternate DELTA disk (594) if you have **no** local service to the macro or copy files. *fm* is the alternate LOCAL disk (295) if you **have** any local service to the macro or copy files.

```
copyfile cpnew exec a = = fm (olddate replace ■
copyfile cpnew maclib a = = fm (olddate replace ■
```

9. To make the MACLIB you just created available to general users, copy it from the CP database to your CMS alternate BUILD1 disk (490):

```
copyfile cpnew maclib origin-fm = = target-fm2 (olddate replace ■
```

   origin-fm is the disk you copied the changed files to in the previous substep. *target-fm* is the CMS alternate BUILD1 disk (490).

10. Erase the working copies on your A-disk:

```
erase cpnew exec a ■
erase cpupdate exec a ■
erase cpnew maclib a ■
```

If you have any image library updates, refer to *VM/XA SP Planning and Administration.*

Step 2. Update the CP Message Repository

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Determine your system default national language:

   ```
   query lang ■
   ```

   langid — See Table 5 on page 311 to identify the language corresponding to *langid*. Note the country code for that language.

3. Invoke the VMFNLS EXEC to update the CP message repository:

   ```
   vmfnls hcpmesy repos 56643089 cp ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   y is the country code for your system national language.

4. If you have updates to the message repositories for any other national languages installed on your system, repeat substep 3 for each language.

5. Copy the changed files from your A-disk to the CP database: *fm* is the alternate DELTA disk (594) if you have **no** local service to the macro or copy files. *fm* is the alternate LOCAL disk (295) if you **have** any local service to the macro or copy files.

   ```
   copyfile hcpmesy {text|txtnnnnn} a = = fm (olddate replace ■
   copyfile hcpmesy listing a = = fm (olddate replace ■
   ```

6. Erase the working copies on your A-disk:

   ```
   erase hcpmesy {text|txtnnnnn} a ■
   erase hcpmesy listing a ■
   ```

Step 3. Assemble the Changed ASSEMBLE Files

Do this step if you have local modifications to ASSEMBLE files.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Determine which ASSEMBLE files you must reassemble:

   - Although you will not receive any updates affecting HCPRIO ASSEMBLE or HCPSYS ASSEMBLE, you must reassemble these files because macros used within these files may have been serviced.
   - Use the list of parts for which you have both local modifications and IBM service that you made in Chapter 13, Applying Service to CP.

3. Copy and unpack the ASSEMBLE files you need to reassemble from the CP current BASE2 disk (394), to your A-disk:

   ```
   copy hcpxxx assemble fm = = a (unpack olddate ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   Substitute the last 3 characters of the filename of the ASSEMBLE file for *xxx*.

4. Use the VMFHASM EXEC to update and assemble all the files for which you have both IBM service and local modifications:

   ```
   vmfhasm hcpxxx 56643089 cp ■
   ```

   Substitute the last 3 characters of the filename of the ASSEMBLE file for *xxx*.

   The text file created will be placed on the A-disk.

5. Copy all the text files from the A-disk to the alternate LOCAL1 (295) disk:

   ```
   copy hcpxxx {text|xxxnnnnn} a = = fm (replace olddate ■
   ```

6. If you reassembled any files that are not in the CP load list, copy the text decks to the alternate LOCAL1 disk again, giving the copy the filetype TEXT:

   ```
   copy hcpxxx {text|xxxnnnnn} a = text fm (replace olddate ■
   ```

7. Erase all the ASSEMBLE files and text files on your A-disk:

   ```
   erase hcpxxx assemble a ■
   erase hcpxxx {text|xxxnnnnn} a ■
   ```

8. Repeat substeps 3 – 7 for each file that must be reassembled.

Step 4. Build the CP Nucleus

Warning:

- **If you have received service for HCPLDL ASSEMBLE or the HCPMDLAT macro, you** must **issue UTILITY CPLOAD to rebuild the CPLOADEXEC before you build the nucleus. See "Step 7. Build Utilities" on page 501 for more details.**
- **If you have received service to HCPLDR ASSEMBLE, you** must **issue UTILITY GEN HCPLDR to rebuild the HCPLDR LOADER.**
- **Before you run VMFBLD, you** must also **rebuild the other CP build lists if they are serviced. The complete list of CP build lists is found in the :BLD. section of the CP portion of the PPF.**

  Use the procedures in "Step 8. Regenerate System Product Interpreter Programs" on page 504 to rebuild the build lists, and return here.

1. Make a backup copy of your system, using the DASD dump restore program.
2. Define storage as 16 megabytes and IPL your system disk:

```
define storage 16m ■
STORAGE =     16M

STORAGE CLEARED - SYSTEM RESET

ipl sysaddr ■
VM/XA CMS 5.6 mm/dd/yy hh:mm

■
Ready; T=n.nn/n.nn hh:mm:ss
```

Defining storage causes a system reset.

sysaddr is your CMS system disk (either 190 or 490).

After receiving the version identification, press **ENTER**.

3. Spool punch output to your own virtual reader:

```
spool punch * ■
Ready; T=n.nn/n.nn hh:mm:ss
```

4. Invoke the VMFBLD EXEC to build a new CP nucleus:

```
vmfbld 56643089 cp (punch ■
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)

RDR FILE fileno SENT FROM MAINT PUN AS fileno RECS nnnn COPY 001 N NOHOLD NOKEEP
```

The system loader punches the CP nucleus. This message tells you that the punch file is complete.

```
Ready; T=n.nn/n.nn hh:mm:ss
```

If you get return code 4, the reason may be that a PTF now being applied has another PTF, outside CP, as a prerequisite. Check the $VMFBLD $ERRLOG file for message DMSBNC1854W.

5. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

6. Ensure that the proper files are in your's virtual reader:

```
query rdr * all ■
```

The ALL operand requests a display of all information about the reader files.

```
ORIGINID FILE CLASS RECORDS  CPY HOLD DATE  TIME     NAME      TYPE     DIST
MAINT    nnnn * PUN nnnnnnnn 001 NONE mm/dd hh:mm:ss $$$TLL$$  IPL      SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

7. Order your reader so that the CMS nucleus will be processed first:

```
order rdr fileno ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fileno is the file number of the CMS nucleus.

8. Ensure that the virtual reader is readied for class * reader files:

```
query virtual 00c ■
RDR  000C CL cl NOCONT NOHOLD   EOF    READY
     000C 2540 CLOSED    NOKEEP
Ready; T=n.nn/n.nn hh:mm:ss
```

9. If the virtual reader is not class *, issue:

```
spool rdr class * keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

The KEEP option places the spool file in USER HOLD status after it has been read. To remove this file from your reader, issue **purge rdr** *fileno*, where *fileno* is the file number of the reader file.

Perform the next two substeps only **if you are generating a CP nucleus with a preferred virtual machine.**

10. Precautions must be taken if the new CP nucleus is to contain a virtual=real area (including virtual=fixed storage) but you are not ready to load (IPL) the new nucleus. If the virtual=real guest is running a job, then let it finish its present work and do not allow more work to begin. The reason for this precaution is that virtual=real recovery may fail in the event of a system restart.

11. Find out how much virtual storage MAINT has and define more if necessary. If you are generating a CP nucleus with a preferred virtual machine, MAINT's virtual storage must be at least 4 megabytes greater than the sum of the VRSIZE, VRFREE, and RIO370 operands in the SYSSTORE macro instruction in HCPSYS ASSEMBLE. For more information on the SYSSTORE macro instruction, refer to *VM/XA SP Planning and Administration.*

```
vmfsetup 56643089 cp ■
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

```
xedit hcpsys assemble ■
:
locate /VRSIZE ■
qquit ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Check the size of the VRSIZE and VRFREE operands.

```
query virtual storage ■
STORAGE = nnnnM
define storage nnnnm ■
STORAGE = nnnnM
STORAGE CLEARED - SYSTEM RESET
```

12. If you are using the starter system defaults, issue the following commands to make sure that the new nucleus does not overlay the production nucleus on the system residence volume while your system is in production:

```
set machine 370 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

```
cpformat 423 xasres 000-endcyl ■
CPFORMAT:
FORMAT WILL ERASE CYLINDERS 0000-0009 ON DISK 423
DO YOU WANT TO CONTINUE ? (YES | NO)
yes ■
FORMAT STARTED
FORMAT COMPLETE
CURRENT ALLOCATION
TYPE CYLINDERS
.....
PERM 0000-endcyl
ENTER ALLOCATION DATA
TYPE CYLINDERS
.....
```

CPFORMAT the 423 minidisk with label XASRES.

The value of *endcyl* depends on two things: the DASD type and whether you are are using the 423 minidisk defined in the sample directory, or a full-pack minidisk that you have defined on a second XASRES volume.

Device Type	*endcyl* **for Sample 423**	*endcyl* **for Full-Pack 423**
3350	0012	0554
3375	0015	0958
3380	0009	0884
3380-E4	0009	1769
3380-K	0009	2654
3390-1	0009	1112
3390-2	0009	2225

```
end ■
CURRENT ALLOCATION
TYPE CYLINDERS
.....
PERM 0000-endcyl
ALLOCATE COMPLETE
VOLUME LABEL IS NOW 'XASRES'
HCPFAM3841 CPFORMAT COMPLETE
SYSTEM RESET
```

Allocate the whole area as PERM.

```
define 123 nnn ■
DASD nnn DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
```

nnn is any address you are not using.

```
define 423 123 ■
DASD 0123 DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
```

Define the 423 minidisk as virtual 123. This is the default address for the system residence volume.

```
define nnn 423 ■
DASD 0423 DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
```

Define the a backup pack for the system residence volume. This is your old 123 minidisk.

13. You must perform this substep if you redefined MAINT's virtual storage as greater than 16 megabytes:

```
set machine xa ■
```

Storage addresses greater than 16 megabytes require 370/XA architecture.

14. Load (IPL) your virtual reader:

```
ipl 00c ■
MSG FROM MAINT   :
  HCPGEN9010W NUCLEUS LOADED ON XASRES
  CP ENTERED; DISABLED WAIT
  PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus is successfully loaded on the system residence device. If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

If you entered a disabled wait state with a code 8028, you did not observe the warning notes dealing with virtual storage size in the substeps above.

15. You have now loaded the nucleus to the 423 disk. (This is also the test 123 minidisk, labeled MNT123.) If you are using the starter system defaults, this is a holding area for the new nucleus.
16. Issue the following commands:

```
set machine 370 ■
define storage 16m ■
```

17. Save the CP load map.
    a. The load map has been spooled to MAINT's virtual printer. To save the load map on MAINT's LOCAL1 (295) minidisk:

```
spool prt nohold ■
close prt ■
ipl sysaddr ■
```

sysaddr is your CMS system disk (either 190 or 490).

    b. Query the printer to see if the load map is still there:

```
query prt all ■
OWNERID  FILE   CLASS RECORDS  CPY HOLD DATE  TIME     DIST
MAINT  fileno A PRT nnnnnnnn 001 NONE mm/dd hh:mm:ss    SYSPROG
Ready; T=n.nn/n.nn hh:mm:ss
```

The CP load map is the file with a blank filename and filetype. It has approximately 30,000 records. The exact size varies according to the system default language, local modifications, and VMSUP level.

CP Load Map in Virtual Printer Only

c. Transfer the printer file to your reader:

```
transfer prt fileno * rdr ■
RDR FILE fileno TRANSFERRED FROM * FILE fileno
0001 FILE TRANSFERRED
Ready; T=n.nn/n.nn hh:mm:ss
```

End of CP Load Map in Virtual Printer Only

d. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cp ■
Ready; T=n.nn/n.nn hh:mm:ss
query rdr * all ■
OWNERID FILE   CLASS RECORDS  CPY HOLD USERFORM OPERFORM KEEP MSG
MAINT   fileno A PRT nnnnnnnn 001 NONE STANDARD STANDARD OFF  OFF
```

Note the *fileno* in this message. You will use it in your next command.

```
receive fileno fn ft fm ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Save the load map on the LOCAL1 (295) minidisk.

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

1) *ymdd* represents the date in four digits: last digit of the year, month (represented as 1−9, A, B, C), and day.

2) Nucleus maps are very large. You may have to save old maps on another disk.

e. To print a copy of the load map for the CP nucleus:

```
print fn ft d ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Substitute the filename and filetype you entered on the RECEIVE command above. The load map is printed on your system printer.

18. Redefine your nucleus areas at their original addresses:

```
define 423 nnn ■
DASD nnn DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
define 123 423 ■
DASD 0423 DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
define nnn 123 ■
DASD 0123 DEFINED
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 5. Test the CP Nucleus

Warning: You can do this step only **if your test XASRES volume is a full-pack minidisk. If you are using a 10-cylinder minidisk, the only testing you can do is verifying that you can load the nucleus on the test 423 minidisk. You did this when you IPLed your test XASRES in Step 4. Go to Step 6.**

Now that you have applied service to all necessary parts, you should test the system to make sure that the problem has been fixed.

1. Test your new nucleus:
   a. Use the IPL DDRXA utility with the DUMP NUC option to dump the nucleus on 123 to tape.
   b. Define your test XASRES volume as virtual 4A0. (If you use the sample HCPRIO in this book, you cannot use address 123 because that address is defined as a console.)

      ```
      detach 0123 ■
      define 423 04A0 ■
      ```

   c. Set your virtual machine in XA mode and define your console at a valid virtual address in the HCPRIO file:

      ```
      set machine xa ■
      Ready; T=n.nn/n.nn hh:mm:ss
      query cons ■
      CONS vdevno ...
      ⋮
      define vdevno 0020 ■
      term conmode 3270 ■
      ```

   d. IPL your test XASRES volume **at second level**. Do **not** perform a hardware IPL.

      ```
      ipl 4A0 clear ■
      ■                                          Press ENTER to create an interrupt.
      ```

   If you can IPL your test XASRES volume successfully, continue with Step 5.

   Note: You may need to define some temporary spool and paging space.
2. Test your system at second level. If it performs to your satisfaction, do "Step 6. Put the New CP System into Production" on page 500. If the system fails, do not do Step 6. Find out what went wrong and start over with a new fix.

3. Shut down your second-level system. A class A user (usually the primary system operator) must issue the SHUTDOWN command.

```
shutdown ■
SYSTEM SHUTDOWN STARTED
SYSTEM WARM START DATA SAVED
SYSTEM SHUTDOWN COMPLETE
HCPGIR450W CP ENTERED; DISABLED WAIT
             PSW 000A0000 00000961
```

4. Link to your production-level XASRES pack in read/write mode:

```
#cp link * 123 123 mr ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Step 6. Put the New CP System into Production

1. Load (IPL) MAINT's virtual reader:

```
ipl 00c clear ■

hh:mm:ss * MSG FROM MAINT    :
  HCPGEN9010W NUCLEUS LOADED ON XASRES
  HCPGIR450W  CP ENTERED; DISABLED WAIT
  PSW 000A0000 00009010
```

The 9010 disabled wait state indicates that the CP nucleus is successfully loaded on the system residence device.

If you enter a disabled wait state with a code 691, you did not observe the warning notes above.

If you receive a different disabled wait state code, see *VM/XA SP System Messages and Codes Reference*.

2. Shut down your system. A class A user (usually the primary system operator) must issue the SHUTDOWN command.

```
shutdown ■
SYSTEM SHUTDOWN STARTED
SYSTEM WARM START DATA SAVED
SYSTEM SHUTDOWN COMPLETE
HCPGIR450W CP ENTERED; DISABLED WAIT
            PSW 000A0000 00000961
```

3. IPL the new CP nucleus:

```
IPL sysadr ■
```

Step 7. Build Utilities

Note: This step shows you how to build utilities. It does **not** show you how to do assemblies. UTILITY EXEC builds utilities. It does **not** assemble anything.

1. IPL CMS:

```
ipl sysadr ■                                  sysadr is the CMS system disk (either 190 or 490).
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
```

2. Establish the appropriate minidisk access order:

```
vmfsetup 56643089 cp ■
```

3. Determine whether you have to build any utilities.
   - Make and sort a list of the files you are servicing:

```
listfile hcp* text fm (exec ■                 fm is the alternate DELTA disk (594).
listfile hcpldr txt* fm (append ■
listfile hcpmdlat * fm (append ■
listfile hcpsadmp * fm (append ■

listfile hcp* text fm (append ■               fm is the alternate LOCAL1 disk (295).

rename cms exec a hcptext = = ■               The exec option saves the list in a file called
xedit hcptext exec ■                          CMS EXEC. Any previously existing CMS
sort * 1 15 ■                                 EXEC on your A-disk is erased.
file ■
```

   - Check this list against Table 13. You must rebuild any utility you find listed there.

4. Find each utility you have to rebuild in the first column of the table. Enter the command in the second column:

```
utility gen filename ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Table 13 (Page 1 of 2). UTILITY GEN Commands

Assemble Filename	Command to Enter	File Created by This Command
Note: If you receive service for a CP TEXT deck not listed in this table, it will be regenerated in some other way. You do not need to regenerate it now. (See Table 16 on page 573 in Appendix A, "VM/XA System Product Regeneration Requirements" for the procedure used. Text decks not listed in this table, or in Table 16 on page 573, will become part of the nucleus when the system is built if they are listed in the CP load list.)		
HCPBSL	**utility gen 3card loader** ■	3CARD LOADER
HCPCCF HCPFA*x* HCPFON	**utility gen cpfmtxa** ■	CPFMTXA MODULE
HCPCCU	**utility gen hcpccu** ■	HCPCCU MODULE

Table 13 (Page 2 of 2). UTILITY GEN Commands

Assemble Filename	Command to Enter	File Created by This Command
HCPDDC HCPDDR HCPDDT HCPDNC HCPDNT	utility gen ddr ■	IPL DDRXA DDR MODULE
HCPDIR	utility gen directxa ■ directxa user direct ■	DIRECTXA MODULE
HCPED*x*	utility gen dumpload ■	DUMPLOAD MODULE
HCPIFC	utility gen cperepxa ■	CPEREPXA MODULE
HCPIMG	utility gen genimage ■	GENIMAGE MODULE
HCPLDL HCPMDLAT	utility cpload ■	CPLOAD EXEC
HCPLDR	utility gen hcpldr ■	HCPLDR MODULE
HCPMOW	utility gen monwrite ■	MONWRITE MODULE
HCPNMT	utility gen imagelib ■	IMAGELIB MODULE
HCPOVE	utility gen override ■	OVERRIDE MODULE
HCPRET	utility gen retrieve ■	RETRIEVE MODULE
HCPSADMP	utility gen hcpsadmp ■	Stand-alone dump program

5. Test the utilities.
6. Copy any utilities generated on the A-disk to the alternate DELTA disk. Then erase them on the A-disk:

```
copyfile fn ft A = = fm ■
erase fn ft A ■
```

fn and *ft* are entries from Table 13 on page 501.
fm is the alternate DELTA disk (594).

7. Copy all serviced MODULES to their appropriate build disks:
   - Make a list of the utilities you are servicing.

     Note: This list includes all serviced utilities on the alternate DELTA disk whether or not you regenerated them in the previous step.

     ```
     filelist * MODULE fm ■
     ```

     fm is the alternate DELTA disk (594).

   - Make a list of all parts for each filename in the filelist by pressing **PF9**. Press **PF9** for each file in the original filelist.
   - Copy the new executable version of the module to the appropriate build disks:
     - If an executable version (with filetype MODULE) exists on the BUILD1 disk, copy the new version there.
     - If an executable version does not reside on one of these disks, refer to the documentation supplied with the PTF to determine where it should reside.

`copyfile / = = fm2 (olddate replace ■`	Enter this command on the filelist screen next to the file you want to copy. You are copying these files from the alternate DELTA disk (594) to the BUILD1 disk (490). Use filemode 2 for files copied to the BUILD1 disk so they can be seen by other users.

- Press **PF2** to refresh the screen to verify that your changes have been made.
- When you have processed each file in the original filelist screen use **PF3** to exit filelist.

8. Repeat substep 7 for the other utilities. Replace the filetype of MODULE with filetypes of DDRXA, LOADER, and EXEC.

Step 8. Regenerate System Product Interpreter Programs

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Regenerate any System Product Interpreter programs (EXECs, XEDIT macros, and the $DASD$ CONSTS foil) for which you have received IBM service and for which you have local modifications. (If you have none, go to substep 3).

   - Use the list of parts for which you have both local modifications and IBM service that you made in Chapter 13, Applying Service to CP.
   - If you reworked or kept your local modifications, you must now regenerate the System Product Interpreter programs (EXECs, XEDIT macros, and the $DASD$ CONSTS file). Repeat the following three commands for each file that needs to be regenerated.

     Note: *ft* is EXEC, XEDIT, or CONSTS. Remember to include the leading dollar sign ($) when it is specified.

`copyfile` *fn* $*ft origin-fm* = = *destination-fm* ■	*origin-fm* is the BASE2 disk (394). *destination-fm* is the alternate DELTA disk (594). The file is copied to the alternate DELTA disk because the EXECUPDT command puts the object file on the same disk as the source file.
`execupdt` *fn ft fm* `(ctl hcpxa hist` *option* ■	HCPXA is the CP control filename. For any files with a SID code in columns 63 – 71, use the SID option.
`erase` *fn* $*ft fm* ■	*fm* is the alternate DELTA disk (594).

3. Copy all serviced EXECs to their appropriate build disks:

   - Make a list of the EXECs you are servicing.

     Note: This list includes all the serviced EXECs on your alternate DELTA disk whether or not you regenerated them in the previous step.

`filelist * EXEC` *fm* ■	*fm* is the alternate DELTA disk (594).

   - Make a list of all parts of each filename in the filelist using **PF9**. Press **PF9** for each file in the original filelist.
   - Identify the latest version of the EXEC:
     - If you just regenerated the EXEC in the previous step, you already have the latest level.
     - If you did not regenerate the EXEC, use the associated AUX files to determine the latest PTF against the given EXEC. This PTF number is the top one in the AUX file. Locate the PTF-numbered version of the EXEC (with filetype EXC*nnnnn*, where *nnnnn* is the PTF number). This PTF-numbered version of the EXEC is the version you want to use.
     - If the EXEC is not maintained with updates, it will not have any associated AUX files and you have to use the internal documentation in the EXEC to determine the PTF-numbered version to

use. **Do not depend on date and time stamps**; examine the actual files. Each file ends with an update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn* and XED*nnnnn* are exact images of the corresponding EXEC and XEDIT executables.

- Copy the PTF-numbered version that you just selected to its executable filetype on the alternate DELTA disk. This step does not apply to those EXECs that you regenerated with the EXECUPDT command.

  `copyfile / = EXEC fm (olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. *fm* is the alternate DELTA disk (594).

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Copy the new executable version of the module to the appropriate build disks:
  - If an executable version (with filetype EXEC) exists on the BUILD1 disk, copy the new version there.
  - If an executable version exists on the TASK disk, copy the new version there.
  - If an executable version does not reside on one of these disks, refer to the documentation provided with the PTF to determine where it should reside.

  `copyfile / = = fm2 (olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. You are copying these files from the alternate DELTA disk (594) to the BUILD1 disk (490), the TASK disk (5E5), or both.

  Use filemode 2 for files copied to the BUILD1 disk so they can be seen by other users.

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Return to the original filelist screen using **PF3**.
- When you have processed each file in the original filelist screen, use **PF3** to exit filelist.

4. Repeat substep 3 for XEDIT macros. Replace EXEC with XEDIT and EXC*nnnnn* with XED*nnnnn*.
5. Repeat substep 3 for the $DASD$ CONSTS file. Replace EXEC with CONSTS and EXC*nnnnn* with CON*nnnnn*.

Step 9. Test and Rebuild CMS

Perform Steps 10, 13, 15, 16, and 17 of Chapter 14, "Rebuilding CMS after Applying Service" on page 431, to make the files on the 490 disk available to users.

Step 10. Back Up the CP System

Use the procedure in Step 31 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" to back up your new CP system.

This is the end of the service procedure for CP.

Chapter 16. Service to DV

This chapter describes:

- Receiving PUT or COR service for DV
- Applying PUT or COR service to DV
- Rebuilding DV.

Should You Be Doing This Now?

- Have you done the preparation steps in Chapter 9, "Preparing For Service" on page 415? If not, do so now.
- Otherwise, continue with this chapter.

Step 1. Receive Service

Note: If you are using ServiceLink to receive service electronically, use the procedure in Chapter 19, "Using ServiceLink to Receive Service" on page 533.

1. Attach a tape drive to MAINT:

   ```
   attach vdevno * 181 ■
   ```

2. Mount and ready the PUT or COR tape.

3. Map the PUT or COR tape and receive the documentation. Mapping the tape means verifying that all the necessary files exist. Enter:

   ```
   release c ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   VMFREC INFO (MEMOS *xxx* loads documents to your C-disk if it is accessed, otherwise to your A-disk.

   ```
   vmfrec info (memos xxx ■
   DMSREC1852I This is n of n
                level level {PUT|COR} tape
   ```

   xxx is **put** if you are mapping a PUT tape, or **cor** if you are mapping a COR tape. If the tape you mounted is not the type of tape you specified, you receive an error message.

   VMFREC INFO (MEMOS maps the tape and loads the tape map, PUT DOCUMENT or COR DOCUMENT, and the *Memo to Users* (56643089 MEMO) to your A-disk. Read these documents before going on. Print these documents to save a record of them. If you do not save a copy of these documents, they will be over-written the next time you receive a PUT or COR tape.

   If your system default language is uppercase American English, you may get message DMSMGM814E. This message is caused by a problem with the REXX date function. You can ignore it.

   ```
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

4. Review the receive exception log ($VMFREC $ERRLOG) to be sure that the MEMOs were loaded correctly:

   ```
   vmfview receive ■
   ```

5. Receive the service files to the target minidisks you specified in the product parameter file. Enter:

   ```
   vmfrec 56643089 dv (xxx ■
   ```

   xxx is **put** for a PUT tape, or **cor** for a COR tape. If you need to invoke an override file, substitute that file's name for **56643089**.

   ```
   DMSREC1852I This is n of n
                level level {PUT|COR} tape
   ```

```
DMSREC1869R 56643089 DV begins on {PUT|COR}
              level volume vol.  Mount
              correct volume and press ENTER
              or type QUIT.
■
DMSREC1852I This is n of n
              level level {PUT|COR} tape
DMSREC1086I The current Service Diskmap contains
              the map of the mounted tape.
DMSREC1804I Receiving service for component DV of product
              56643089
:
Ready; T=n.nn/n.nn hh:mm:ss
```

If you do not have the correct volume mounted, mount it now.

The VMFREC EXEC loads the service files to the target minidisk.

6. Review the receive exception log ($VMFREC $ERRLOG), and correct any problems before continuing:

```
vmfview receive ■
Ready; T=n.nn/n.nn hh:mm:ss
```

To Receive More Than One COR Tape

If you want to receive additional COR tapes at this time:

1. You must suppress the automatic merging that occurs before a receive. Read "Preventing Automatic Merging" in Appendix G, "Controlling Disk String Merges" on page 861 and choose a method of preventing automatic merging.
2. For each COR tape you want to receive, repeat substeps 2 – 6.
3. Create an apply list for the VMFAPPLY EXEC to use that contains the PTFs from all the COR tapes you just received:

```
listfile hcsxa $apc* fm ■
HCSXA    $APCALL  fm
HCSXA    $APC0109 fm
HCSXA    $APC0119 fm
HCSXA    $APC0126 fm
HCSXA    $APC9C22 fm
```

fm is the alternate DELTA disk (593).

```
xedit hcsxa $applist fm ■
get $apcymdd ■
file ■
```

Edit the HCSXA $APPLIST file on the alternate DELTA disk. This file will already contain the apply list from the last tape that you loaded. Do a GET command in XEDIT once for each HCSXA $APC*ymdd* file shown by your LISTFILE command.

Note: If the VMFAPPLY EXEC encounters duplicate entries for the same PTF, it issues a message telling you that the PTF has already been applied. This is an informational message; you can ignore it.

4. Create an exclude list for the VMFAPPLY EXEC to use that contains the excluded PTFs from all the COR tapes you just received:

Note: Do not be concerned if the exclude lists are empty.

```
listfile hcsxa $exc* fm ■
HCSXA    $EXCALL  fm
HCSXA    $EXC0109 fm
HCSXA    $EXC0119 fm
HCSXA    $EXC0126 fm
HCSXA    $EXC9C22 fm
```

fm is the alternate DELTA disk (593).

```
xedit hcsxa $exclist fm ■
get $excymdd ■
file ■
```

Edit the HCSXA $EXCLIST file on the alternate DELTA disk. This file will already contain the exclude list from the last tape that you loaded. Do a GET command in XEDIT once for each HCSXA $EXC*ymdd* file shown by your LISTFILE command.

End of To Receive More Than One COR Tape

Step 2. Apply the Updates

1. Consult your IBM service representative for the appropriate preventive service planning (PSP) bucket for your system's VMSUP level and PUT level. Check the bucket and add the appropriate entries to the exclude list. The exclude list is called HCSXA $EXCLIST.

   Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prerequisite for the second PTF.

   If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

2. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 dv ■
   ```

3. Apply the service, that is, create auxiliary control files:

   ```
   vmfapply 56643089 dv (xxx ■
   DMSAPP1853I Processing PTF ptfnum
   :
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   xxx is **put** for PUT service, or **cor** for COR service.

4. Review the apply exception log ($VMFAPP $ERRLOG), and correct any problems before continuing. The information in the apply exception log is used later when you rebuild the dump viewing facility.

   ```
   vmfview apply ■
   ```

   If you receive message number DMSAPP1885W, VMFAPPLY has applied service to a module for which you have a local modification.

   You can use the PF keys with the VMFVIEW EXEC to get useful lists of information. If you position the cursor on a line that contains message 1885W and press **PF2**, VMFVIEW displays all the occurrences of message 1885W. Similarly, if you press **PF6**, VMFVIEW displays all the messages with the BD: prefix. You can use the XEDIT PUT command to save these messages in a file for future reference.

   Decide what you want to do with the local modification:

   - To remove a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543, and Chapter 22, "Removing Service from VM/XA SP" on page 559.
   - To rework a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543.
   - To keep a local modification, do nothing now. You will regenerate the part to pick up the local modification when you rebuild DV.

   Save the names of all the parts for which you have both local modifications and IBM service. You will need the part names later when you rebuild DV.

   Whether you remove, rework, or keep local modifications, you do not have to reissue VMFAPPLY.

Step 3. Build the Dump Viewing Facility

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 dv ■
   ```

2. Build the dump viewing facility:

   ```
   utility dvfgend all ■
   ```

3. Copy all MODULE files created by UTILITY DVFGEND ALL to the BUILD1 disk (490), then erase them on your A-disk:

   ```
   copy fn module a = = fm2 (OLDDATE REPLACE
   Ready; T=n.nn/n.nn hh:mm:ss
   erase fn module a ■
   ```

4. Copy all MAP files created by UTILITY DVFGEND ALL to the APPLY disk (692), then erase them on your A-disk:

   ```
   copy fn map a = = fm ■
   Ready; T=n.nn/n.nn hh:mm:ss
   erase fn map a ■
   ```

5. Copy DVF interface files to the CMS system disk:

   ```
   vmfbld 56643089 dv ■
   ```

6. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

   ```
   vmfview build ■
   ```

Step 4. Regenerate System Product Interpreter Programs

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 dv ■
   ```

2. Regenerate any EXECs and XEDIT macros for which you have received IBM service and for which you have local modifications. (If you have none, go to substep 3).

   - Use the list of parts for which you have both local modifications and IBM service that you made in "Step 2. Apply the Updates."
   - If you reworked or kept your local modifications, you must now regenerate the EXECs and XEDIT macros. Repeat the following three commands for each file that needs to be regenerated.

     Note: *ft* is either EXEC or XEDIT. Remember to include the leading dollar sign ($) when it is specified.

     `copyfile` *fn* `$`*ft origin-fm = = destination-fm* ■
     origin-fm is the BASE2 disk (393). *destination-fm* is the alternate DELTA disk (593).

     The file is copied to the alternate DELTA disk because the EXECUPDT command puts the object file on the same disk as the source file.

     `execupdt` *fn ft fm* `(ctl hcsxa hist` *option* ■
     HCSXA is the dump viewing facility control filename.

     For any files with a SID code in columns 63 – 71, use the SID option.

     `erase` *fn* `$`*ft fm* ■
     fm is the alternate DELTA disk (593).

3. Copy all serviced EXECs to their appropriate build disks:

   - Create a list of the EXECs you are servicing.

     Note: This list includes all the serviced EXECs on your alternate DELTA disk whether or not you regenerated them in the previous step.

     `filelist * EXEC` *fm* ■
     fm is the alternate DELTA disk (593).

   - Create a list of all parts of each filename in the filelist using **PF9**. Press **PF9** for each file in the original filelist.
   - Identify the latest version of the EXEC:
     - If you just regenerated the EXEC in the previous step, you already have the latest level.
     - If you did not regenerate the EXEC, use the associated AUX files to determine the latest PTF against the given EXEC. This PTF number is the top one in the AUX file. Locate the PTF-numbered version of the EXEC (with filetype EXC*nnnnn*, where *nnnnn* is the PTF number). This PTF-numbered version of the EXEC is the version you want to use.
     - If the EXEC is not maintained with updates, it will not have any associated AUX files and you have to use the internal documentation in the EXEC to determine the PTF-numbered version to use. **Do not depend on date and time stamps**; examine the actual files. Each file ends with an

update summary that indicates the file's service history. The latest version is the one with the most service. Files with filetypes of EXC*nnnnn* and XED*nnnnn* are exact images of the corresponding EXEC and XEDIT executables.

- Copy the PTF-numbered version that you just selected to its executable filetype on the alternate DELTA disk. This step does not apply to those EXECs that you regenerated with the EXECUPDT command.

  `copyfile / = EXEC` *fm* `(olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. *fm* is the alternate DELTA disk (593).

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Copy the new executable version of the module to the appropriate build disks:
  - If an executable version (with filetype EXEC) exists on the BUILD1 disk, copy the new version there.
  - If an executable version exists on the TASK disk, copy the new version there.
  - If an executable version does not reside on one of these disks, refer to the documentation supplied with the PTF to determine where it should reside.

  `copyfile / = =` *fm*`2 (olddate replace` ■

  Enter this command on the filelist screen next to the file you want to copy. You are copying these files from the alternate DELTA disk (593) to the BUILD1 disk (490), the TASK disk (5E5), or both.

  Use filemode 2 for files copied to the BUILD1 disk so they can be seen by other users.

- Press **PF2** to refresh the screen to verify that your changes have been made.
- Return to the original filelist screen using **PF3**.
- When you have processed each file in the original filelist screen, use **PF3** to exit filelist.

4. Repeat substep 3 for XEDIT macros. Replace EXEC with XEDIT and EXC*nnnnn* with XED*nnnnn*.

Step 5. Test and Rebuild CMS

Perform Steps 10, 13, 15, 16, and 17 of Chapter 14, "Rebuilding CMS after Applying Service" on page 431, to make the files on the 490 disk available to users.

Step 6. Back Up the Dump Viewing Facility

Use the procedure in Step 31 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" to back up the dump viewing facility.

This is the end of the service procedure for the dump viewing facility.

Chapter 17. Service to GCS

This chapter describes:

- Receiving PUT or COR service for GCS
- Applying PUT or COR service to GCS
- Rebuilding GCS.

Should You Be Doing This Now?

- Have you done the preparation steps in Chapter 9, "Preparing For Service" on page 415? If not, do so now.
- Otherwise, continue with this chapter.

Step 1. Receive Service

| **Note:** If you are using ServiceLink to receive service electronically, use the procedure in Chapter 19, "Using ServiceLink to Receive Service" on page 533.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 gcs ■
   ```

2. Attach a tape drive to MAINT:

   ```
   attach vdevno * 181 ■
   ```

| 3. Mount and ready the PUT or COR tape.

4. Map the PUT or COR tape and receive the documentation. Mapping the tape means verifying that all the necessary files exist. Enter:

   ```
   release c ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   VMFREC INFO (MEMOS *xxx* loads documents to your C-disk if it is accessed, otherwise to your A-disk.

   ```
   vmfrec info (memos xxx ■
   DMSREC1852I This is n of n,
                level level {PUT|COR} tape
   ```

   xxx is **put** if you are mapping a PUT tape, or **cor** if you are mapping a COR tape. If the tape you mounted is not the type of tape you specified, you receive an error message.

   VMFREC INFO (MEMOS maps the tape and loads the tape map, PUT DOCUMENT or COR DOCUMENT, and the *Memo to Users* (56643089 MEMO) to your A-disk. Read these documents before going on. Print the documents to save a record of them. If you do not save a copy of these documents, they will be over-written the next time you receive a PUT or COR tape.

   If your system default language is uppercase American English, you may get message DMSMGM814E. This message is caused by a problem with the REXX date function. You can ignore it.

   ```
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

| 5. Review the receive exception log ($VMFREC $ERRLOG) to be sure that the MEMOs were loaded correctly:

   ```
   vmfview receive ■
   ```

6. Receive the service files to the target minidisks you specified in the product parameter file. Enter:

   ```
   vmfrec 56643089 gcs (xxx ■
   ```

   xxx is **put** for a PUT tape, or **cor** for a COR tape. If you need to invoke an override file, substitute that file's name for **56643089**.

```
DMSREC1852I This is n of n,
             level level {PUT|COR} tape
DMSREC1806I The current Service Diskmap contains
             the map of the mounted tape
DMSREC1869R 56643089 GCS begins on {PUT|COR}
             level volume vol.  Mount
             correct volume and press ENTER
             or type QUIT.
■
DMSREC1852I This is n of n,
             level level {PUT|COR} tape
DMSREC1806I The current Service Diskmap contains
             the map of the mounted tape
DMSREC1804I Receiving service for component GCS
             of product 56643089
⋮
Ready; T=n.nn/n.nn hh:mm:ss
```

If you do not have the correct volume mounted, mount it now.

The VMFREC EXEC loads the service files to the target minidisk.

7. Review the receive exception log ($VMFREC $ERRLOG), and correct any problems before continuing:

```
vmfview receive ■
Ready; T=n.nn/n.nn hh:mm:ss
```

To Receive More Than One COR Tape

If you want to receive additional COR tapes at this time:

1. You must suppress the automatic merging that occurs before a receive. Read "Preventing Automatic Merging" in Appendix G, "Controlling Disk String Merges" on page 861 and choose a method of preventing automatic merging.
2. For each COR tape you want to receive, repeat substeps 3 – 7.
3. Create an apply list for the VMFAPPLY EXEC to use that contains the PTFs from all the COR tapes you just received:

```
listfile csixa $apc* fm ■
CSIXA    $APCALL  fm
CSIXA    $APC0109 fm
CSIXA    $APC0119 fm
CSIXA    $APC0126 fm
CSIXA    $APC9C22 fm
```

fm is the alternate DELTA disk (896).

```
xedit csixa $applist fm ■
get $apcymdd ■
file ■
```

Edit the CSIXA $APPLIST file on the alternate DELTA disk. This file will already contain the apply list from the last tape that you loaded. Do a GET command in XEDIT once for each CSIXA $APC*ymdd* file shown by your LISTFILE command.

Note: If the VMFAPPLY EXEC encounters duplicate entries for the same PTF, it issues a message telling you that the PTF has already been applied. This is an informational message; you can ignore it.

4. Create an exclude list for the VMFAPPLY EXEC to use that contains the excluded PTFs from all the COR tapes you just received:

   Note: Do not be concerned if the exclude lists are empty.

```
listfile csixa $exc* fm ■
CSIXA    $EXCALL  fm
CSIXA    $EXC0109 fm
CSIXA    $EXC0119 fm
CSIXA    $EXC0126 fm
CSIXA    $EXC9C22 fm
```

fm is the alternate DELTA disk (896).

```
xedit csixa $exclist fm ■
get $excymdd ■
file ■
```

Edit the CSIXA $EXCLIST file on the alternate DELTA disk. This file will already contain the exclude list from the last tape that you loaded. Do a GET command in XEDIT once for each CSIXA $EXC*ymdd* file shown by your LISTFILE command.

End of To Receive More Than One COR Tape

Step 2. Apply the Updates

1. Consult your IBM service representative for the appropriate preventive service planning (PSP) bucket for your system's VMSUP level and PUT level. Check the bucket and add the appropriate entries to the exclude list. The exclude list is called CSIXA $EXCLIST.

   Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prerequisite for the second PTF.

   If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

2. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 gcs ■
   ```

3. Apply the service, that is, create auxiliary control files:

   ```
   vmfapply 56643089 gcs (xxx ■
   DMSAPP1853I Processing PTF ptfnum
   ⋮
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   xxx is **put** for PUT service, or **cor** for COR service.

4. Review the apply exception log ($VMFAPP $ERRLOG), and correct any problems before continuing. The information in the apply exception log is used later when you rebuild the dump viewing facility.

   ```
   vmfview apply ■
   ```

   If you receive message number DMSAPP1885W, VMFAPPLY has applied service to a module for which you have a local modification.

   You can use the PF keys with the VMFVIEW EXEC to get useful lists of information. If you position the cursor on a line that contains message 1885W and press **PF2**, VMFVIEW displays all the occurrences of message 1885W. Similarly, if you press **PF6**, VMFVIEW displays all the messages with the BD: prefix. You can use the XEDIT PUT command to save these messages in a file for future reference.

   Decide what you want to do with the local modification:

   - To remove a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543, and Chapter 22, "Removing Service from VM/XA SP" on page 559.
   - To rework a local modification, see Chapter 20, "Receiving and Applying Local Service" on page 543.
   - To keep a local modification, do nothing now. You will regenerate the part to pick up the local modification when you rebuild GCS.

   Save the names of all the parts for which you have both local modifications and IBM service. You will need the part names later when you rebuild GCS.

   Whether you remove, rework, or keep local modifications, you do not have to reissue VMFAPPLY.

Step 3. Build a New GCS Macro Library

> **Note:** This step describes a method of updating the GCS macro libraries that isolates all of the serviced macros and control blocks into a separate MACLIB using the VMFMAC EXEC. The new MACLIB is added to the MACS record in the CSIXA CNTRL file.
>
> An alternate method of updating the GCS macro libraries is to update each affected MACLIB with the changed macro or control block by using the MACLIB command. (See *VM/XA SP CMS Command Reference* for details on the MACLIB command.)
>
> If you choose the alternate method of building new GCS macro libraries, do so now and continue with "Step 4. Update the GCS Message Repository" on page 525.

Note: If you want to rebuild the entire CSISP MACLIB, you must have the GCS SOURCE feature tape.

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 gcs ■
   ```

2. Determine the macros and control blocks you have service for:

   - Search the DELTA string for all macros and control blocks you have service for:

`listfile * A*CI` *fm1* `(exec args fname ■`	*fm1* is the alternate DELTA (896) disk.
`listfile * A*CI` *fm2* `(append args fname ■`	*fm2* is the intermediate alternate DELTA (791) disk.
`listfile * A*CI` *fm3* `(append args fname ■`	*fm3* is the current DELTA (596) disk.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.
   - Save the filenames in the GCSUPDAT EXEC and erase the CMS EXEC:

     ```
     copyfile cms exec a gcsupdat = = (replace ■
     erase cms exec a ■
     ```

   - Determine which of the updated files are macros (filetype MACRO), and which are control blocks (filetype COPY). Store the results in the GCSNEW EXEC:

     ```
     gcsupdat listfile % macro * (append args ftype ■
     gcsupdat listfile % copy * (append args ftype ■
     ```

     Ignore the messages that appear on your screen while these two commands are processing.

   - If a CMS EXEC does not exist on your A-disk after these commands, no serviced macros or control blocks were found on the DELTA string. Continue with substep 3.
   - Save the filenames in the GCSNEW EXEC:

     ```
     copyfile cms exec a gcsnew = = (replace ■
     ```

- Edit the GCSNEW EXEC to remove any duplicate entries:

```
xedit gcsnew exec ■
sort * 1 15 ■
:
file ■
```

3. Determine the macros and control blocks for which you have local service. Follow your installation's naming conventions for locally serviced files. Locally serviced files should reside on the LOCAL1 disk string.

   - If no macros and control blocks have local service, continue with substep 4.
   - Otherwise, add your locally serviced macros and control blocks to GCSNEW EXEC. When you are done, the GCSNEW EXEC should list all macros and control blocks for which you have received IBM service (if any), plus any macros and control blocks for which you have local service.

     If you do not have a GCSNEW EXEC on your A-disk, copy the CSIGPI EXEC to use as a template:

     ```
     copyfile csigpi exec fm gcsnew = a (recfm f lrecl 80 ■
     ```

     If you created the GCSNEW EXEC by copying the CSIGPI EXEC, use the contents of the CSIGPI EXEC only for a guide to the format of the entries. After you have added the macro and control block names you need, in the proper format, delete the macro names that originated from the CSIGPI EXEC:

     ```
     xedit gcsnew exec a ■
     :
     file ■
     ```

4. If you have no updated macros or control files at this point, continue with "Step 4. Update the GCS Message Repository" on page 525.

5. Generate the new macro library on your A-disk (191) by using the VMFMAC EXEC:

   ```
   vmfmac gcsnew csixa ■
   DMSUPD178I Updating macroname MACRO fm
   DMSUPD178I Applying macroname update fm
   macroname MACRO ADDED
   :
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

6. The contents of GCSNEW COPY were appended to GCSNEW MACLIB. You can now erase GCSNEW COPY (a file created by VMFMAC) on your A-disk (191):

   ```
   erase gcsnew copy a ■
   ```

7. Add the new macro library name to the TEXT MACS card in the CSIXA CNTRL file.

   List the new macro library first so that you use the updated macros. If you have already done this during previous service to GCS, you do not have to do this now.

   ```
   xedit csixa cntrl fm ■
   locate/TEXT MACS ■
   TEXT MACS maclib ...
   change/MACS/MACS GCSNEW/ ■
   TEXT MACS GCSNEW maclib ...
   file ■
   ```

   fm is the GCS BASE1 disk (595).

8. Copy the changed files from your A-disk to the GCS database: *fm* is the alternate DELTA disk (896) if you have **no** local service to the macros or control blocks. *fm* is the alternate LOCAL disk (495) if you **have** any local service to the macros or control blocks.

   ```
   copyfile cmsnew exec a = = fm (olddate replace ■
   copyfile cmsnew maclib a = = fm (olddate replace ■
   ```

9. To make the MACLIB you just created available to general users, copy it from the GCS database to your GCS alternate BUILD1 disk (895):

   ```
   copyfile gcsnew maclib origin-fm = = target-fm2 (olddate replace ■
   ```

   origin-fm is the disk you copied the changed files to in the previous substep. *target-fm* is the GCS alternate BUILD1 disk (895).

10. Erase the working copies on your A-disk:

    ```
    erase gcsnew exec a ■
    erase gcsupdat exec a ■
    erase gcsnew maclib a ■
    ```

Step 4. Update the GCS Message Repository

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 gcs ■
   ```

2. Determine your system default national language:

   ```
   query lang ■
   ```

   langid — See Table 5 on page 311 to identify the language corresponding to *langid*. Note the country code for that language.

3. Invoke the VMFNLS EXEC to update the GCS message repository:

   ```
   vmfnls csimesy repos 56643089 gcs ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   y is the country code for your system national language.

4. If you have updates to the message repositories for any other national languages installed on your system, repeat substep 3 for each language.

5. Copy the changed files from your A-disk to the GCS database: *fm* is the alternate DELTA disk (896) if you have **no** local service to the macro or copy files. *fm* is the alternate LOCAL disk (495) if you **have** any local service to the macro or copy files.

   ```
   copyfile csimesy {text|txtnnnnn} a = = fm (olddate replace ■
   copyfile csimesy listing a = = fm (olddate replace ■
   ```

6. Erase the working copies on your A-disk:

   ```
   erase dmscsiy {text|txtnnnnn} a ■
   erase dmscsiy listing a ■
   ```

Step 5. Assemble and Build the GCS Nucleus

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 gcs ■
   ```

2. If you did not create the TESTNSS EXEC in "Step 9. Create Test EXECs and Files" on page 451 of Chapter 14, "Rebuilding CMS after Applying Service," do so now.

3. Define a new named saved system with an alternate name:

   ```
   testnss altsysname ■
   ```

   altsysname is an alternate name for your GCS system (for example, ALTGCS).

   ```
   HCPNSD440I The Named Saved System (NSS) altsysname
              was successfully defined
              in fileid fileno.
   OWNERID  FILE TYPE CL RECS DATE  TIME     FILENAME   FILETYPE ORIGINID
   *NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMS        NSS      MAINT
   *NSS     nnnn NSS  A  nnnn mm/dd hh:mm:ss CMSXA      NSS      MAINT
   ⋮
   *NSS     nnnn NSS  S  nnnn mm/dd hh:mm:ss altsysname NSS      MAINT
   ```

4. Create a configuration file for the GCS system you are servicing:

   ```
   copyfile gcs group fm altsysname = = (replace ■
   xedit altsysname GROUP ■
   locate /SYSNAME= ■
   change /SYSNAME=sysname/SYSNAME=altsysname/ ■
   locate /SEGMENTNAME= ■
   change /SEGMENTNAME=sysname/SEGMENTNAME=altsysname/ ■
   file ■
   ```

   fm is the GCS LOCAL1 disk (495).

5. Create an ASSEMBLE file for the GCS system you are servicing:

   ```
   copyfile altsysname group fm = assemble = (replace ■
   ```

   fm is the GCS LOCAL1 disk (495).

6. Use the VMFHASM EXEC to update and assemble the ASSEMBLE file:

   ```
   vmfhasm altsysname 56643089 gcs ■
   ```

   ```
   DMSUPD181E No update files were found
   ASSEMBLING altsysname
   *** altsysname TEXT CREATED! ***
   PRT FILE fileno SENT FROM MAINT PRT AS fileno
      RECS nnnn COPY 001 A NOHOLD NOKEEP
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   These are informational messages. You can ignore them.

7. Copy the TEXT file from your A-disk to the alternate LOCAL1 disk:

   ```
   copy altsysname text a = = fm (replace ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   fm is the GCS LOCAL1 disk (495).

8. Issue the following commands:

```
change rdr class n class m ■
Ready; T=n.nn/n.nn hh:mm:ss
spool pun * class n nocont noeof ■
Ready; T=n.nn/n.nn hh:mm:ss
spool rdr class n keep ■
Ready; T=n.nn/n.nn hh:mm:ss
close pun ■
Ready; T=n.nn/n.nn hh:mm:ss
```

9. XEDIT the GCSLOAD EXEC so that it will pick up the text deck you created in substep 6 on page 526 for the system you are now servicing:

```
xedit gcsload EXEC ■
```

```
locate /&3 GCS ■
change /&3 GCS/&3 altsysname/ ■
file ■
```

If you have already edited GCSLOAD EXEC to service another GCS system, you must locate and change the *altsysname* for that system instead of GCS.

10. Invoke VMFBLD to build the GCS nucleus:

```
vmfbld 56643089 GCS (punch ■
```

```
vmfbld 56643089 altsysname (punch ■
```

Note: Use this command if you created a PPF override file to maintain more than one GCS system.

```
0000001 FILE CHANGED
LOAD LIST: $$$TLL$$ EXEC A1 (MNT191)
RDR FILE filenol SENT FROM MAINT PUN AS fileno2
   RECS nnnn COPY 001 N NOHOLD NOKEEP
```

11. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

```
vmfview build ■
```

12. Issue the following commands:

```
spool pun off eof ■
Ready; T=n.nn/n.nn hh:mm:ss
close rdr ■
Ready; T=n.nn/n.nn hh:mm:ss
spool rdr class n keep ■
Ready; T=n.nn/n.nn hh:mm:ss
```

13. IPL your reader to save the new GCS system in the named saved system you created:

```
ipl 00c clear ■
MSG FROM MAINT: CSIIN134I altsysname has nnnnnn
  bytes of available common free storage
HCPNSD440I The Named Saved System (NSS) altsysname
            was successfully defined in fileid fileno
PRT FILE fileno1 SENT FROM MAINT RDR AS fileno2
   RECS nnnn COPY 001 N NOHOLD NOKEEP
System reset
```

This printer file is the GCS load map. Note the second file number.

14. IPL CMS (or ALTCMS):

```
ipl {cms|altcms} ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

15. Save and print the GCS load map:

```
#cp transfer prt fileno2 to * rdr ■
vmfsetup 56643089 gcs ■
receive fileno2 gcsnuc map fm ■
```

fm is the GCS LOCAL1 disk (495).

You can assign any filename and filetype to the load map. You may want to adopt a naming convention for the load maps, to save the map from each build. For example, you can name each map in the format *compname*map *ymdd* to distinguish each build.

Notes:

a. *ymdd* represents the date in four digits: last digit of the year, month (represented as 1−9, A, B, C), and day.

b. Nucleus maps are very large. You may have to save old maps on another disk.

Once you have the load map saved on disk, you can print a copy of it by issuing the following command:

```
print gcsnuc map fn ■
```

fm is the GCS LOCAL1 disk (495).

Maintaining More Than One GCS System

16. You can maintain multiple GCS systems using the technique described below for each GCS system. Each GCS system must have a unique name and must be targeted to a different BUILD disk.

To use different BUILD disks (BUILD*n*) for each nucleus, you should create an override to the product parameter file for each nucleus you are maintaining. In the override file, change the BUILD*n* target. Each override file should have a unique filename and a filetype of **PPF**.

A sample override file follows.

```
* This is the alias override for the VM/XA System Product 56643089
:OVERLST.  altsysname CORaltsysname
*
:altsysname.    GCS 56643089    * PUT Service
:BLD.
GCSLOAD VMFBDNUC BUILDn
:END.
*
:CORaltsysname. GCS 56643089    * COR Service
:MERGE.    APPLYX DELTAX
:MDA.      UPDATE
APPLYX     892 491
DELTAX     896 791
:BLD.
GCSLOAD VMFBDNUC BUILDn
:END.
```

End of Maintaining More Than One GCS System

Step 6. Test and Rebuild CMS

Perform Steps 10, 13, 15, 16, and 17 of Chapter 14, "Rebuilding CMS after Applying Service" on page 431, to make the files that VMFBLD EXEC loaded to the 490 disk in Step 5 available to users.

Step 7. Test and Rebuild the GCS Nucleus

Test the GCS nucleus you just built. When you are satisfied, repeat Step 5 to rebuild the nucleus with its original name.

Step 8. Back Up the GCS Named Saved Systems

Use the procedure in Step 30 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)" to back up the GCS named saved systems.

This is the end of the service procedure for GCS.

Chapter 18. Program Update Service to Licensed Programs

This chapter describes how to apply program update service to licensed programs that have a product service EXEC rather than a product parameter file.

To apply service to licensed programs, refer to the publications for the specific licensed program. If the licensed program has a product parameter file, the general concepts described in this book for program update service are applicable. To apply corrective service, see the publications for the specific licensed program.

If the licensed program has a product service EXEC, the following general procedure is applicable.

1. Attach a tape drive to MAINT:

   ```
   attach vdevno * 181 ■
   ```

2. Mount and ready the PUT tape.

3. Map the PUT tape and receive the documentation. Mapping the tape means verifying that all the necessary files exist. Enter:

   ```
   release c ■
   ```

   VMFREC INFO (MEMOS xxx loads documents to the C-disk if it is accessed, otherwise to the A-disk. You want them loaded to the A-disk.

   ```
   vmfrec info (memos put ■
   DMSREC1852I This is n of n,
               level level PUT tape
   ```

   If the tape you mounted is not a PUT tape, you receive an error message.

   VMFREC INFO (MEMOS maps the tape and loads the tape map, PUT DOCUMENT, and the *Memo to Users* (56643089 MEMO) to your A-disk. Read these documents before going on. Print the documents to save a record of them. If you do not save a copy of these documents, they will be over-written the next time you receive a PUT tape.

   If your system default language is uppercase American English, you may get message DMSMGM814E. This message is caused by a problem with the REXX date function. You can ignore it.

   ```
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

4. Review the receive exception log ($VMFREC $ERRLOG) to be sure that the MEMOs were loaded correctly:

   ```
   vmfview receive ■
   ```

5. Receive the service files to the target minidisks you specified in the product parameter file. Enter:

   `vmfrec prodid parameters` ■

   VMFREC invokes the product service EXEC to receive and apply service. You can pass parameters to the product service EXEC.

6. Review the receive exception log ($VMFREC $ERRLOG), and correct any problems before continuing:

   ```
   vmfview receive ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

7. Check the Program Directory for the licensed product (not the VM/XA SP Program Directory) or other licensed program documentation to see whether the product service EXEC, which VMFREC invoked for you in step {RAPROD}, builds the licensed program. For most products, it does. If it does not, invoke VMFBLD to build the licensed program:

   `vmfbld prodid parameters` ■

   VMFBLD invokes the product service EXEC with the BUILD parameter. The product service EXEC then rebuilds any part of the licensed program changed by service that requires a build function. You can pass parameters to the product service EXEC.

8. Review the build exception log ($VMFBLD $ERRLOG), and correct any problems before continuing:

   ```
   vmfview build ■
   ```

Note: For compatibility, the VMSERV EXEC continues to be available to service this type of licensed program.

Chapter 19. Using ServiceLink to Receive Service

Before You Begin:

This chapter is for VM/XA System Product Release 2.1 users who have the IBMLink and ServiceLink products. ServiceLink is a part of IBMLink that lets you access IBM software service information online. You must be authorized to use IBMLink and ServiceLink. In addition, you must be authorized to order service and receive it electronically using ServiceLink. The information in this chapter must be used in conjunction with the *ServiceLink User's Guide*, SH52-0300. The *ServiceLink User's Guide* provides information on retrieving and decompacting the envelope file on your A-disk. When this process is completed, continue with "Step 1. Load the Documentation Envelope" on page 539 for the procedures to receive the service from ServiceLink.

The ServiceLink Envelope File

When you receive service through ServiceLink, the storage medium for the service is a CMS disk file rather than tape. This CMS disk file, called an envelope file, contains other (one or more) disk files that can be further separated into groups of files for processing. The following terminology is used in this chapter:

Envelope File — A CMS disk file that contains the individual files that are to be loaded or otherwise processed.

File (or Disk File) — The individual files that are contained within the envelope file. Each file within the envelope consists of a header record, which contains information to identify the file, and one or more data records containing the contents of the original file.

Header Record — A special record in the envelope file that contains the file identifier for the file, date/time stamp information, and other information needed to load the file.

Data Records — The actual data from the file in the format used by the CMS COPYFILE command.

Group Separator — A special record in the envelope file that marks the boundary between groups of files.

End of Disk — The logical end of the contents of the envelope file. This condition occurs when two successive group separator records are encountered. If the envelope does not have two group separator records at the physical end of the envelope, VMFPLCD will process the file as if they exist.

Envelope File Description

There are two basic types of envelope files shipped by ServiceLink: documentation envelopes and the envelope containing the actual fixes. Additionally, each envelope maybe in two different formats and is processed by different commands, VMFREC and VMFPLCD. Following are examples of the different types of envelope files that may be received from ServiceLink and the command used to process it.

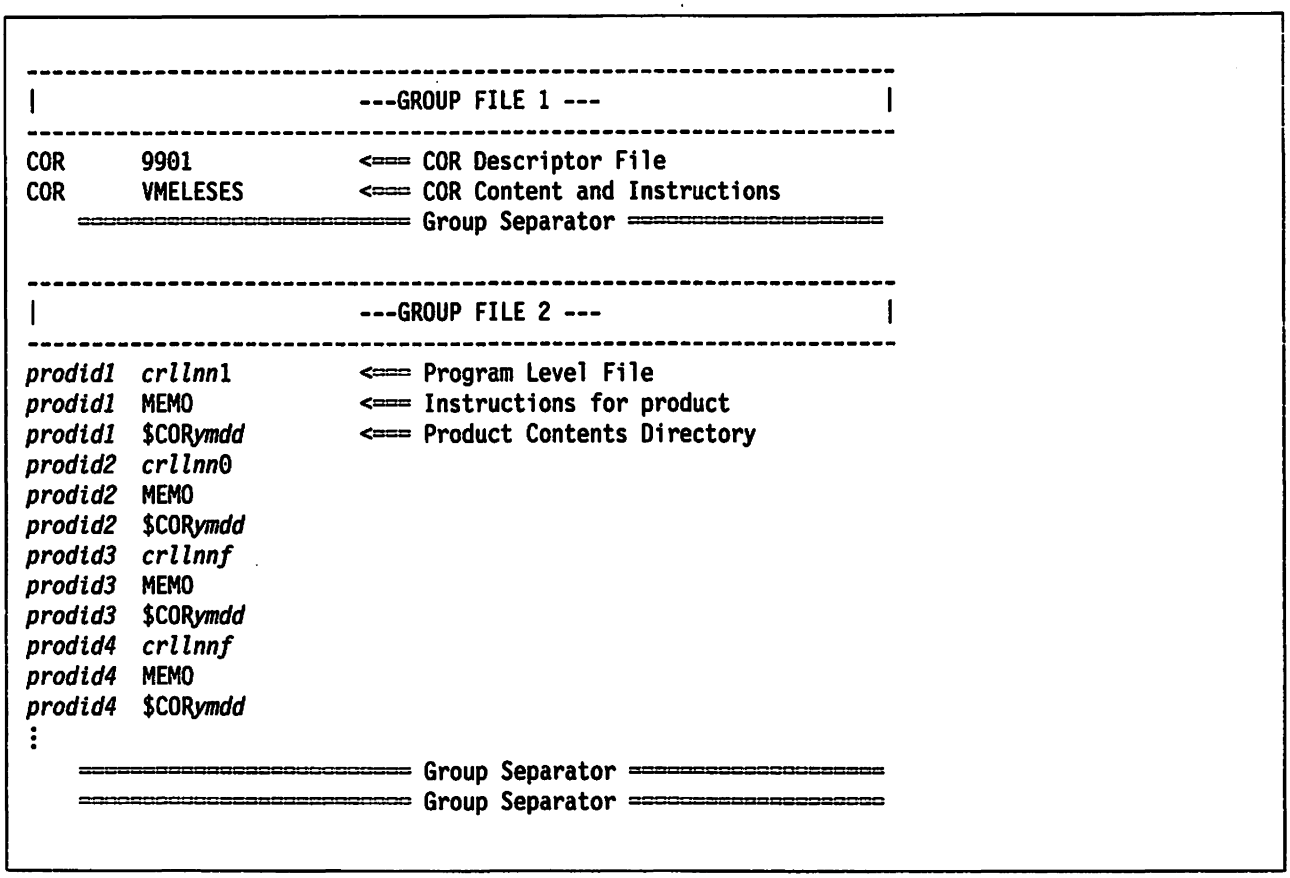


Figure 30. Example of an Envelope File in VMFREC Format. This envelope file has a filetype of VLSTnnnn and contains service documentation.

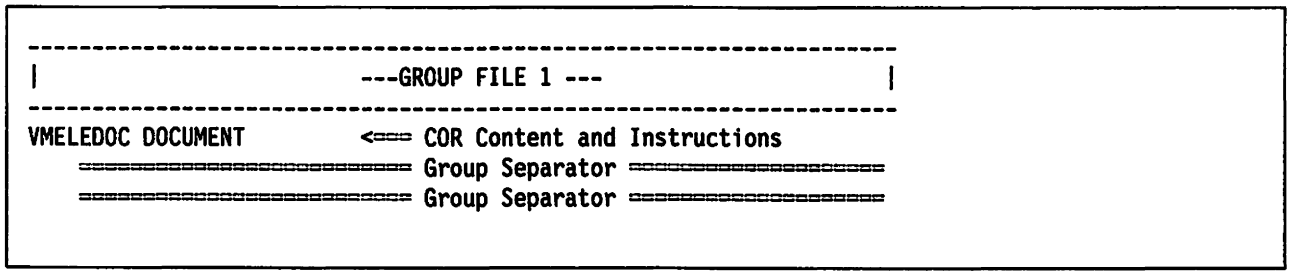


Figure 31. Example of an Envelope File in VMFPLCD Format. This envelope file has a filetype of VLSTnnnn and contains service documentation.

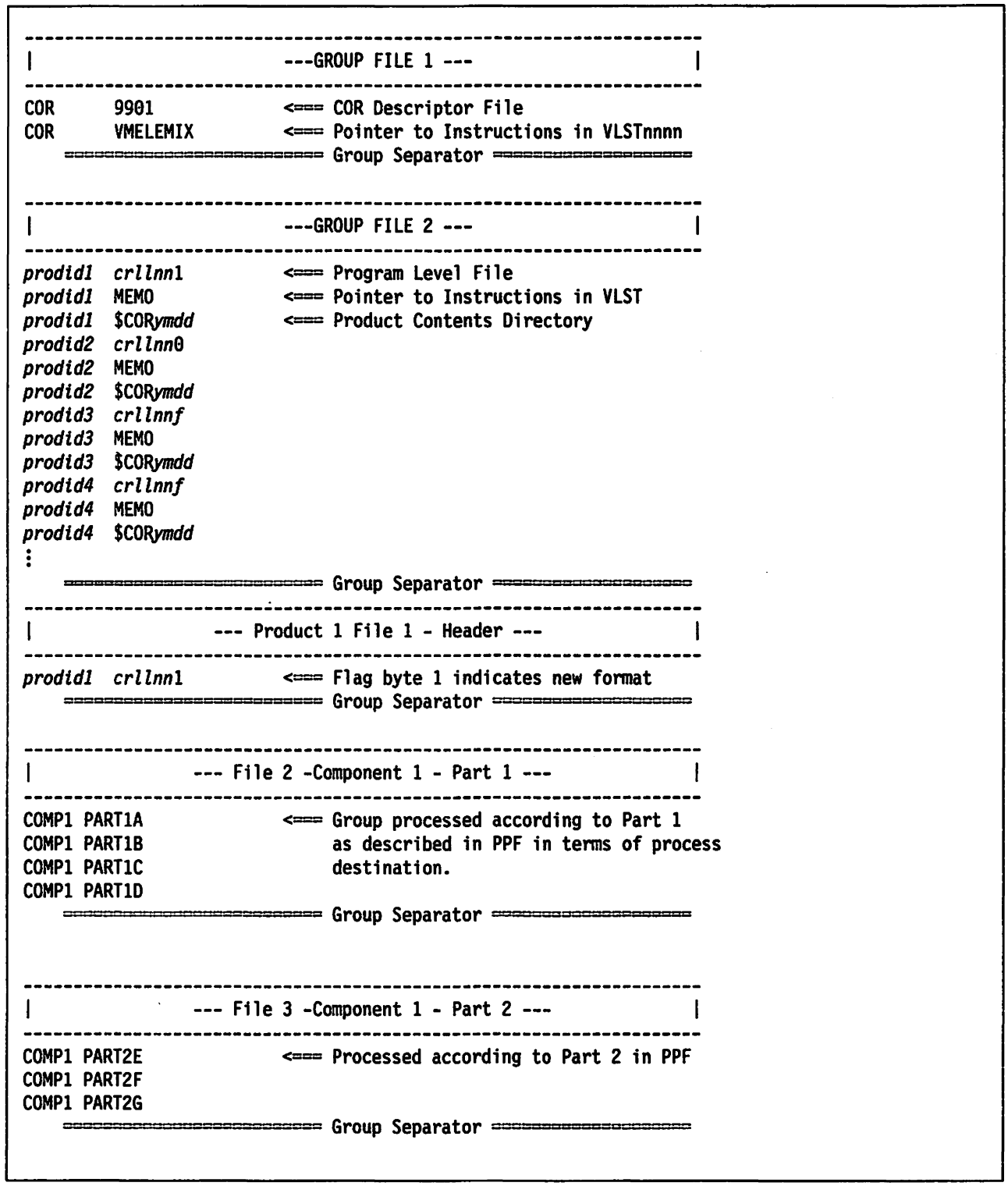


Figure 32 (Part 1 of 2). Example of an Envelope File in VMFREC Format

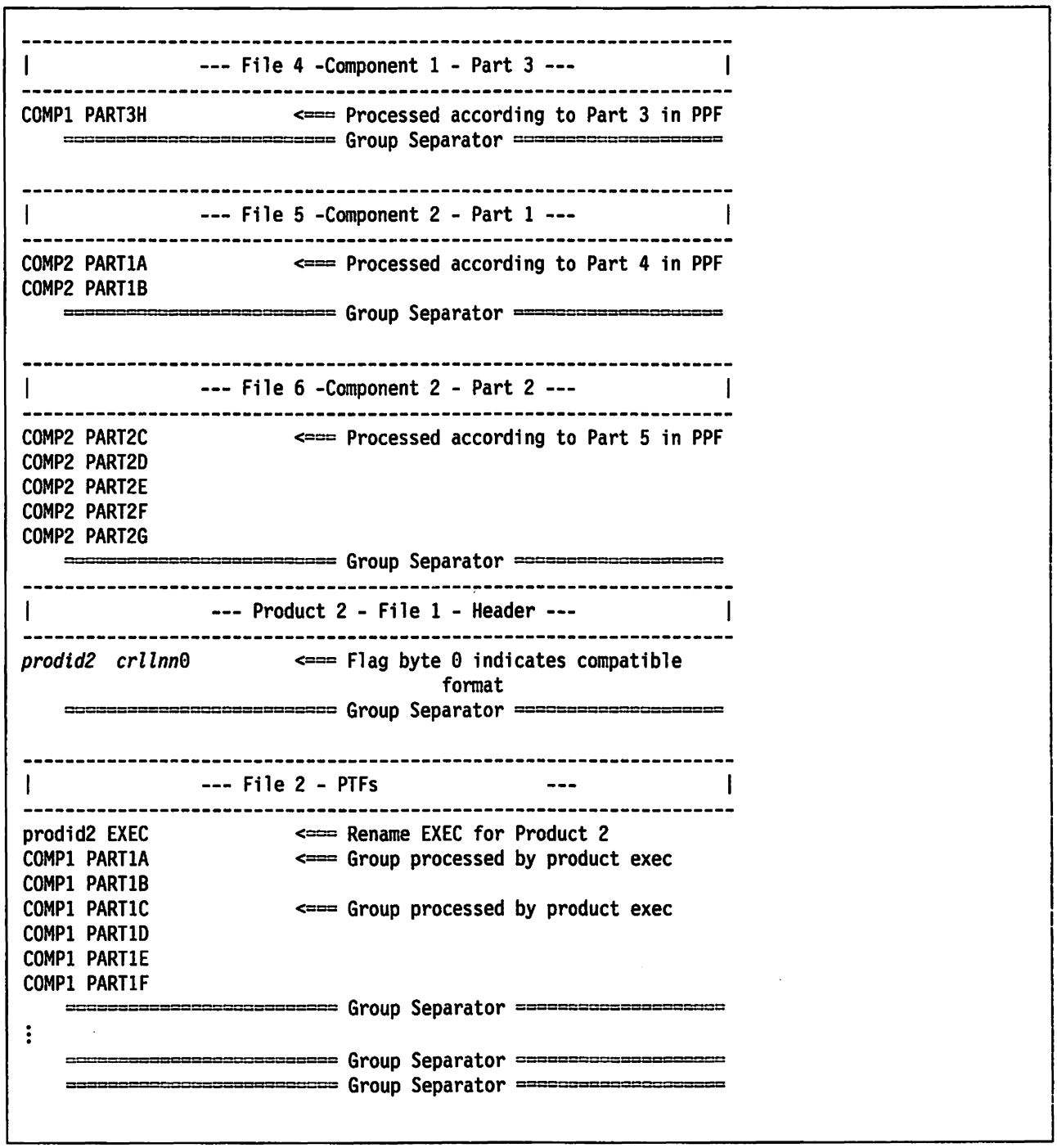


Figure 32 (Part 2 of 2). Example of an Envelope File in VMFREC Format

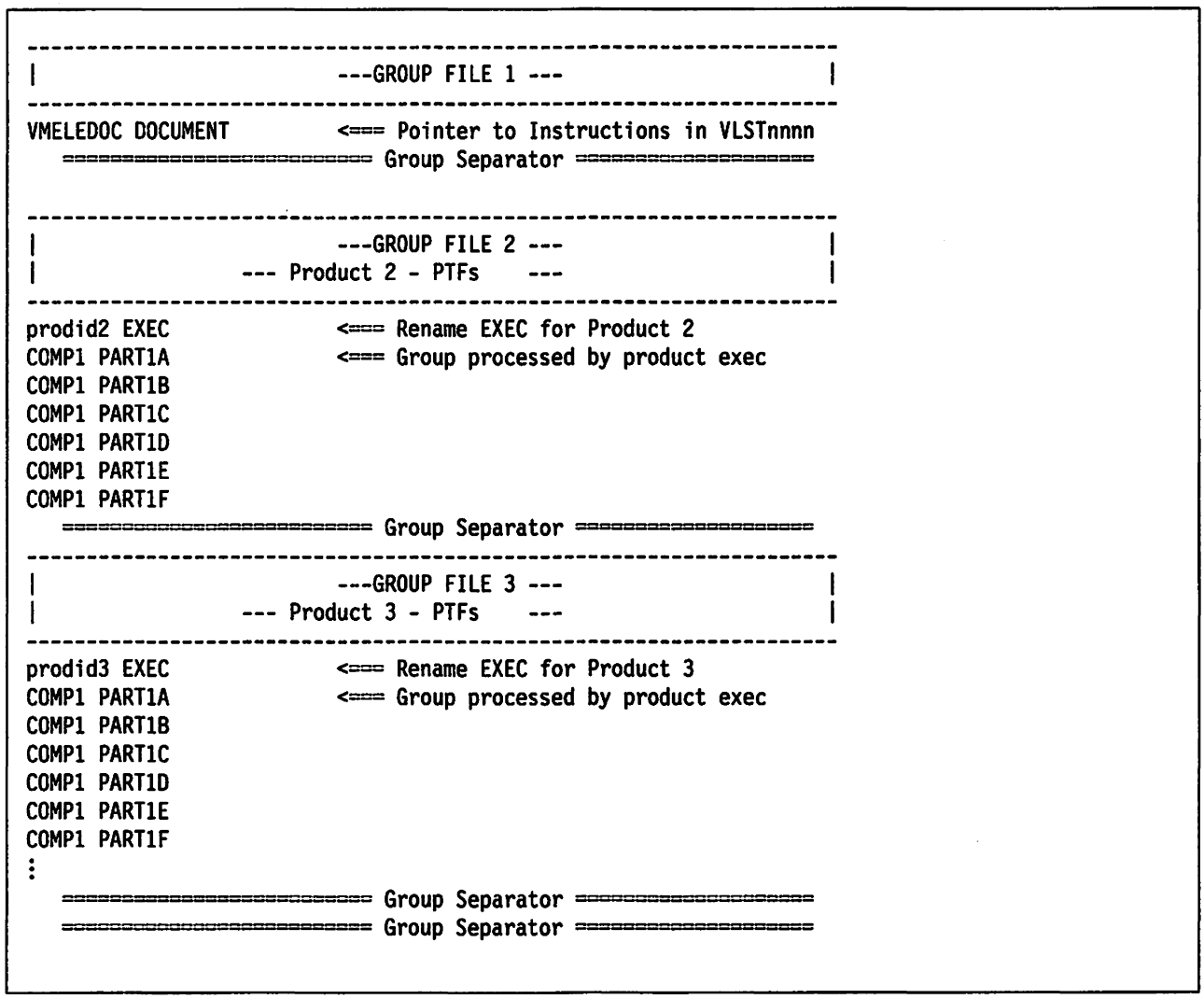


Figure 33. Example of an Envelope File in VMFPLCD Format. This envelope file has a filetype of VPTFnnnn and contains the actual fixes for two products.

CRLLNNF - C = COREQ Flag
R = Release Level of product or SCP
LL= PTF Level of Product or PLC level SCP
NN= Total number of Group Separators for this Product
F = 1 indicates new format, 0 indicates compatible format

Figure 34. Definition of a Program Level Filetype. The CRLLNNF filetype is used to determine the type of receive processing that is required for the product. A 1 in the F field indicates that the product is supported by VMFREC.

Tools Used With ServiceLink

To receive service with ServiceLink, you use one or more of the following EXECs:

- For envelopes in VMFREC format, use the VMFREC EXEC. The format of the VMFREC EXEC is described in "VMFREC EXEC" on page 691.
- For envelopes in VMFPLCD format, use the VMFPLCD EXEC. The VMFPLCD EXEC reproduces the functions of the VMFPLC2 EXEC, except that the storage medium is disk rather than tape. The commands, functions, parameters, and options resemble the VMFPLC2 EXEC as closely as possible. The format of the VMFPLCD EXEC is described in "VMFPLCD EXEC" on page 681.

An additional EXEC, the VMFPLC EXEC, is provided to minimize impacts to existing code currently using VMFPLC2. The VMFPLC EXEC calls either the VMFPLC2 EXEC or the VMFPLCD EXEC depending on the routing in effect. The default routing of commands from VMFPLC is to the VMFPLC2 EXEC. You can still invoke VMFPLC2 or VMFPLCD directly from the command line as well.

The format of the VMFPLC EXEC is described in "VMFPLC EXEC" on page 679. The format of the VMFPLC2 EXEC is described in "VMFPLC2 Command" on page 687.

Step 1. Load the Documentation Envelope

1. Follow the procedures in the *ServiceLink User's Guide* to retrieve and uncompact the envelope file on your A-disk.

   Note: If the filetype of your VLST*nnnn* and VPTF*nnnn* envelope files is not SERVLINK, retain the filename but change the filetype to SERVLINK.

2. Use XEDIT to edit the documentation envelope you have on your A-disk, and check the filename of *fn* MEMOS. If the filename is VMELESES or VMELEMIX, continue with "Step 2. Receive the Service Envelope (VM/XA System Product Release 2.1 Only)" on page 540. If the filename is anything else, continue with "Step 3. Receive the Service Envelope (Other Program Products)" on page 541.

```
x vlstnnnn servlink ■
```

Step 2. Receive the Service Envelope (VM/XA System Product Release 2.1 Only)

1. Receive the service documentation relating to PTF files from the VLST*nnnn* SERVLINK envelope:

   Note: You only need to do this operation once for each envelope file, no matter how many products or components are in each envelope file.

   `vmfrec info env vlst`*nnnn* `(memos COR` ■

   The type of service you are receiving is considered to be COR service.

   VMFREC EXEC maps the envelope file (verifies that the necessary files exist) and creates a SERVICE DISKMAP file on filemode C (or A), loads the COR VMELESES or COR VMELEMIX documentation file to filemode C (or A), and loads the *prodid* MEMO files in the second group to filemode C (or A).

   Continue with substep 2.

2. Read the service file document to determine the products you want to service. Read the *Memo to Users* for each product you plan to service.

3. Receive the service envelope:

   `vmfrec 56643089` *component* `env vtpf`*nnnn* `(cor` ■

   compname is the component you are receiving service for. The type of service you are receiving is considered to be corrective service.

4. Review the receive exception log (VMFREC $ERRLOG), and correct any problems before continuing:

   ```
   vmfview receive ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

5. When you are satisfied that the service has been received correctly, erase the $VMFREC $HISTORY and $VMFREC $ERRLOG files on your A-disk.

6. Continue with the service process:

   - For CMS, continue with "What to Do Next" on page 421.
   - For CP, continue with "What to Do Next" on page 426.
   - For DV, continue with "Step 2. Apply the Updates" on page 511.
   - For GCS, continue with "Step 2. Apply the Updates" on page 521.

Step 3. Receive the Service Envelope (Other Program Products)

1. Reset to the start of the envelope and load the documentation envelope:

   ```
   vmfplcd rst env= vlstnnnn servlink ■
   vmfplcd load * * a env= vlstnnnn servlink (eod ■
   ```

2. Read the service file document to determine the products you want to service. Read the *Memo to Users* for each product you plan to service.

3. Receive the service for the product on a work disk. Refer to your product documentation for procedures to follow for installing corrective service.

   ```
   vmfplcd rst env= vptfnnnn servlink ■
   vmfplcd scan prodid exec (eod ■
   vmfplcd load * * fm (disk ■
   ```

4. To continue with the service process, consult the documentation provided with the product.

Chapter 20. Receiving and Applying Local Service

This chapter describes step-by-step procedures for applying local service to all components of VM/XA System Product using update files.

Introduction

Local service is defined as any service that is applied to your VM/XA SP system that was not supplied by IBM on a COR tape, COR tape envelope (in SES format), or PUT tape.

Warning: The application of local service can be a complicated and error-prone procedure because of the many variables that are involved. IBM strongly advises its customers to order service on a COR tape through the IBM Support Center whenever possible. When it is absolutely necessary to apply service from IBM before it is available on a COR tape, or a local modification is required to tailor a system environment, you must apply the service locally. This should be done very carefully.

Note: Before attempting to apply local service, please read Chapter 6, "VM/XA System Product Service—An Overview," Chapter 7, "How VM/XA System Product Uses Control Files and Update Files," and Chapter 8, "Files Used in PUT and COR Service" of this book. It is very important to have a good understanding of how the service process works before you proceed with this chapter.

Local service can be placed in two categories:

- IBM local service includes any service that you receive from IBM that is not COR closed, such as emergency service from the change team. When a severe problem arises and you cannot wait for a COR tape, you can get emergency service from the change team. This service can be sent to you on a tape or read to you over the phone. In either case, the service is **not** in the format required by the VMFREC and VMFAPPLY EXECs. To receive and apply this service, you **must** do it manually using the instructions in this chapter.
- Customer local service includes any service that is **not** supplied by IBM. Customer local service is used to tailor your VM/XA SP system. An example of customer local service is given in Step 17 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)." The example given shows how to tailor the HCPBOX ASSEMBLE file. To receive and apply this service, you **must** do it manually using the instructions in this chapter.

The instructions in this chapter apply to all components of the VM/XA SP system (CMS, CP, DV, and GCS). Whenever there are variations for different components, they are noted.

Step 1. Preparation

Before you service your VM/XA System Product Release 2.1 system, you should create an alternate level of build disks. Doing so isolates the changed system so it can be tested before you make it available to your general user community.

1. Use the DASD DUMP Restore program to copy the current BUILD1 disk to its corresponding alternate disk:

```
ddr ■
VM/XA SYSTEM PRODUCT DASD DUMP RESTORE PROGRAM
ENTER:
```

```
sysprint cons ■
ENTER:
```

This command tells DDR to send program messages to your console.

```
input 190 devtype MNT190 ■
ENTER:
```

190 is your current CMS system disk. *devtype* is the device type of the DASD volume where 190 is located.

```
output 490 devtype MNT490 ■
ENTER:
```

490 is your alternate CMS system disk. *devtype* is the device type of the DASD volume where 490 is located.

```
copy 000 endcyl ■
```

The value for *endcyl* depends on the device type of your 190 disk:

Device Type	*endcyl*
3350	73
3375	112
3380	71
3390	67

```
DMKDDR711R Volid Read is MNT190.  Do you
             wish to continue?
yes ■
ENTER NEXT EXTENT OR NULL LINE:
■
⋮
END OF COPY
ENTER:
■
END OF JOB
```

You have not yet changed the label of the 490 minidisk. (You will do so in substep 2.) This prompt will not always appear.

2. Re-label the 490 minidisk:

```
access 490 t ■
Ready; T=n.nn/n.nn hh:mm:ss
format 490 t (label ■
ENTER LABEL:
mnt490 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

Access 490 *after* the system disk. You *must* use the **label** option. If you do not, you will erase all the files on the 490 minidisk.

3. Repeat substeps 1 and 2 to copy the current HELP disks (19D, 19C, 19B...) to their corresponding alternate disks (49D, etc.) and to re-label the alternate disks as MNT49*n*. Check the minidisk definitions in your user directory to find the appropriate value for *endcyl*.

Note: There is one HELP disk for each national language supported on your system. If you support only one national language, you only have to do this substep once.

Step 2. Preparation for CMS Local Service Only

In order to be consistent with the philosophy of not building on top of your production system, you must perform the following steps before applying local service to CMS:

1. Copy the latest version of the DMSNGP ASSEMBLE file to the 395 disk. (If there is already a copy of DMSNGP ASSEMBLE or DMSNGP TEXT on the 395 disk, rename it first.) Edit the copy and make the following changes:
   a. Change SYSDISK=190 to SYSDISK=490.
   b. Change IPLADDR=190 to IPLADDR=490.
   c. Change HELP=19D to HELP=49D (or HELP=19C to HELP=49C).
   d. Change SYSNAME=CMS to SYSNAME=*newname*, where *newname* is the name you choose for your test system. The sample procedure in Chapter 14, "Rebuilding CMS after Applying Service," uses ALTCMS.
   e. Change INSTSEG=CMSINST to INSTSEG=*newname*, where *newname* is the name you choose for your test installation segment. The sample procedure in Chapter 14, "Rebuilding CMS after Applying Service," uses ALTINST.
   f. Change the VERSION= and INSTID= parameters to identify your test system.
   g. Make sure that SAVESYS=NO. If SAVESYS=YES, when you IPL CMS the system will try to save the alternate CMS segment that you define in Step 10 of Chapter 14, "Rebuilding CMS after Applying Service" in the existing CMS segment.

These changes cause the new CMS nucleus (with service) to be built on the 490 minidisk.

Step 3. Receive Service

1. Log on to MAINT and establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 compid ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   Substitute for *compid* the component you are servicing (CMS, CP, DV, or GCS).

2. Place the service that you receive in one or more CMS files on the alternate LOCAL1 minidisk.

Service Supplied on Magnetic Tape or Tape Envelope

If service is supplied on a magnetic tape, or in a tape envelope which you receive electronically, use VMFPLC:

```
vmfplc load * * fm(eot disk ■
```

fm is the alternate LOCAL1 disk for the component you are servicing.

End of Service Supplied on Magnetic Tape or Tape Envelope

Service Supplied as Hard Copy or by Telephone

3. If service is supplied as hard copy or by telephone as the result of an emergency, your IBM Service Representative should supply you with the filename and filetype of each required file. Make a list of these names. You must now type in these files. Use the XEDIT command to create each CMS file on the alternate LOCAL1 minidisk.

End of Service Supplied as Hard Copy or by Telephone

Step 4. Apply Service

Changes to the system are provided as new parts, replacement parts, or updates to existing parts initially shipped on the product tape. The following table shows the base filetypes that might be serviced and how they are serviced (update or replacement):

Table 14. Types of Files That Can Be Serviced

Filetype	Corresponding File	Type of Service
TEXT	With corresponding ASSEMBLE	Update
	Without corresponding ASSEMBLE	Replacement
EXEC	With corresponding $EXEC	Update
	Without corresponding $EXEC	Replacement
XEDIT	With corresponding $XEDIT	Update
	Without corresponding $XEDIT	Replacement
MACRO		Update
COPY		Update
Any other types		Replacement

For the following examples, assume that you have received emergency service from the change team. You have received one PTF, but it is not COR closed. A PTF is considered COR closed when it is made available for distribution on a COR tape. The PTF (UM34567) hits two modules, HCPABC and HCPXYZ. Also assume that you are creating customer local service for HCPQRS. These examples are for CP files, but the procedures for all components are the same.

For the list of files that you are servicing, refer to the tape or disk map from the last step if you received service from tape or envelope, or to the list that your IBM Service Representative gave you if you received service as hard copy or over the telephone.

1. Determine the base filetypes of HCPABC, HCPXYZ and HCPQRS.

   For each unique filename that you are servicing, issue the LISTFILE command. Using Table 14 and "VM/XA SP Filetypes and Abbreviations" on page 863 as a guide, write down what the base filetype is for each filename being serviced.

```
listfile hcpabc * * ■
HCPABC   A98765HP  fm
HCPABC   AUXLCL    fm
HCPABC   L00001LC  fm
HCPABC   LCL00001  fm
HCPABC   AUXXA     fm
HCPABC   A12345HP  fm
HCPABC   TXT32123  fm
HCPABC   ASSEMBLE  fm
HCPABC   TEXT      fm
Ready; T=n.nn/n.nn hh:mm:ss
```

   In this example, you can see both an ASSEMBLE file and a corresponding TEXT file. This indicates that HCPABC is serviced by updates (source-maintained).

```
listfile hcpxyz * * ■
HCPXYZ   TXT34567  fm
HCPXYZ   AUXXA     fm
HCPXYZ   A12345PP  fm
HCPXYZ   TXT32123  fm
HCPXYZ   TEXT      fm
Ready; T=n.nn/n.nn hh:mm:ss
```

In this example, there is no corresponding ASSEMBLE file, so the base filetype is TEXT. This indicates that HCPXYZ is serviced by replacement (object-maintained). Do not be confused by the update file HCPXYC A12345PP. It is only an update shell. Update shells are explained below.

```
listfile hcpqrs * * ■
HCPQRS   EXEC      fm
HCPQRS   $EXEC     fm
Ready; T=n.nn/n.nn hh:mm:ss
```

In this example, there is an EXEC and a corresponding $EXEC. This indicates that HCPQRS is source update serviced.

2. Use XEDIT to create an AUX file entry on the AUXLCL level.

   There are three fields that make up an AUX entry. The first field contains the filetype of the update file. Imbedded in this filetype is the 5-digit numeric portion of the APAR number. When you originate an update, this 5-digit number should be a local tracking number. The second field is the level of the fix. For local service, use LCL in this field. If you want to keep your IBM local service separate from your customer local service, you can use LC*x* in this field (*x* is any alphanumeric character). The third field is the PTF number or a local tracking number.

   a. If an AUXLCL file already exists, as in the case of HCPABC, copy it to the alternate LOCAL1.

```
copyfile hcpabc auxlcl fm1 = = fm2 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

   Substitute for *fm1* the filemode of the minidisk on which the AUXLCL file currently resides. *fm2* is the LOCAL1 minidisk.

   The existing HCPABC AUXLCL file has only one entry:

```
L00001LC LCL LC00001 * CUSTOMER LOCAL SERVICE TO CIRCUMVENT A PROBLEM
```

   This is a locally originated fix. 00001 is your local tracking number.

   If the new fix from IBM fixes the problem you circumvented with the entry in the AUXLCL file, replace the entry in the AUXLCL file with the new one from IBM. Your new AUX entry in the AUXLCL file is then:

```
A98765HP LCL UM34567 * IBM UPDATE TO FIX THE PROBLEM
```

   If the new fix from IBM is unrelated to the entry that is in the AUXLCL file, you **must** review your previous customer local service and rework it if necessary. Put the AUX entry for the new fix below the existing AUXLCL entry in the file. Then use XEDIT with the CTL option to rework the old AUX entry. The new AUXLCL file would look like this:

```
L00001LC LCL LC00001 * REWORKED CUSTOMER LOCAL SERVICE
A98765HP LCL UM34567 * IBM UPDATE TO FIX A DIFFERENT PROBLEM
```

   b. If an AUXLCL file does not exist but another AUX file does (as in the case of HCPXYZ), you can copy and rename an existing AUX file to the alternate LOCAL1 and use it as a guide:

```
copyfile hcpxyz auxxa fm1 = auxlcl fm2 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

   Substitute for *fm1* the filemode of the minidisk that the AUXXA file currently resides on. *fm2* is the LOCAL1 minidisk.

For HCPXYZ, you have a replacement text deck. Use XEDIT to look at the self-documenting information at the top of the new text deck:

xedit hcpxyz txt34567 *fm* ■

The system displays a file like the one in Figure 35. *fm* is the LOCAL1 minidisk.

```
A12345PP 201 UM32123 * THIS IS AN OLD FIX
*       HCPXYZ A12345PP fm nnnnnn mm/dd/yy hh:mm:ss
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A98765PP xxx UM34567 * THIS IS THE NEW FIX
*       HCPXYZ A98765PP fm nnnnnn mm/dd/yy hh:mm:ss
* PREREQ: VM12345
* CO-REQ: NONE
* IF-REQ: NONE
:
```

Figure 35. A Sample Replacement Text Deck (HCPXYZ TXT34567)

In AUX files, the most recent entry is on top. In text decks, the most recent entry is on the bottom of the prolog. The first entry in the text deck is already in the HCPXYZ AUXXA file. The second entry is the new fix and requires that an AUX entry be made on the AUXLCL level. The AUX entry can be copied directly from the replacement text deck. The only change should be in the second field. As mentioned above, this field should be LCL, so the AUX entry in HCPXYZ AUXLCL should be:

```
A98765PP LCL UM34567 * THIS IS THE NEW FIX
```

c. If no AUX files exist, as in the case of HCPQRS, use XEDIT to create one on the alternate LOCAL1.

For HCPQRS, there are no existing updates, so when you have finished editing the AUXLCL file, the only entry in it should be the update that you are creating:

```
L00002LC LCL LC00002 * LOCAL FIX TO CIRCUMVENT A PROBLEM
```

Note: This process would be the same for other parts, including the filetypes ASSEMBLE, $EXEC, $XEDIT, MACRO, and COPY.

3. Create an update file or update shell.

   a. For HCPABC, IBM shipped you the update file for the fix.

   b. For HCPXYZ, IBM shipped you only a replacement text deck named HCPXYZ TXT34567. For this part, you must create an update shell. An update shell takes the place of the source update file that would exist if the part were on update service. Just as with the AUXLCL file for HCPXYZ, you can take the information for the update shell from the replacement text deck (see Figure 35). Your update shell for HCPXYZ should look like Figure 36.

```
./ * PREREQ: VM12345
./ * CO-REQ: NONE
./ * IF-REQ: NONE
./ * DEPEND: NONE
./ * * THIS IS AN UPDATE SHELL
```

Figure 36. A Sample Update Shell (HCPXYZ A98765PP)

c. For HCPQRS, you must create your own update file because it is originated locally. Normally, you would already know what kind of change you wanted to make to HCPQRS. To create the update file, you will invoke XEDIT with the CTL option:

```
xedit hcpqrs $exec fm (ctl hcpxa ■
:
file ■
Ready; T=n.nn/n.nn hh:mm:ss
```

If you receive message DMSXUP178I, previous updates are being applied before your new update is created. You should receive message DMSXUP180W. This message tells you that your new update cannot be found. This is because you are in the process of creating it. At this point you can make your updates. Remember that these updates do **not** go into the $EXEC file, but instead are put in the HCPQRS L00002LC update file.

Note: This process is the same for other parts, including files with the filetypes ASSEMBLE, $EXEC, $XEDIT, MACRO, and COPY. The CNTRL file varies from component to component. See Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361 for a description of the CNTRL files and to find out which one is appropriate for the service you are performing.

4. Create executable parts.

   At this point, you should have a correct AUXLCL file and a corresponding update file or shell.

   a. For HCPABC, you still need a text deck. This text deck is generated when you assemble HCPABC during the rebuilding process after you have finished this chapter.

   b. For HCPXYZ, IBM supplied you with a replacement text deck. The text deck, however, is not named consistently with the AUXLCL entry that you created. You must now rename this text deck:

```
rename hcpxyz txt34567 fm = lcl34567 = ■
Ready; T=n.nn/n.nn hh:mm:ss
```

   c. For HCPQRS, you still need an EXEC. This EXEC is generated when you run EXECUPDT for HCPQRS during the rebuilding process, after you have finished this chapter.

   The following lists the base filetypes that may be included in your local service, along with a brief description of how the executable part is generated.

   - ASSEMBLE with a corresponding TEXT or TXT*nnnnn*: **Update**

     You reassemble this part during the rebuilding process.

   - TEXT or TXT*nnnnn* with no corresponding ASSEMBLE: **Replacement**

     You must rename the new TXT*nnnnn* file to LCL*nnnnn*, where *nnnnn* is the PTF number:

```
rename filename txtnnnnn fm = lclnnnnn = ■
Ready; T=n.nn/n.nn hh:mm:ss
```

   - EXEC with a corresponding $EXEC: **Update**

     You will use EXECUPDT to generate the EXEC during the re-building process.

- EXEC with no corresponding $EXEC: **Replacement**

  If you did not receive a new file with the filetype EXEC with your service, and you did receive a file with the filetype of EXC*nnnnn*, where *nnnnn* is the PTF number, this file must be copied with the filetype of EXEC:

```
copyfile filename excnnnnn fm = exec = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

- XEDIT with a corresponding $XEDIT: **Update**

  You will use EXECUPDT to generate the XEDIT macro during the rebuilding process.

- XEDIT with no corresponding $XEDIT: **Replacement**

  If you did not receive a new file with the filetype XEDIT with your service, and you did receive a file with the filetype of XED*nnnnn*, where *nnnnn* is the PTF number, this file must be copied with the filetype of XEDIT:

```
copyfile filename xednnnnn fm = xedit = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

- MACRO: **Update**

  You will use VMFMAC to generate a MACLIB with the new MACRO during the rebuilding process.

- COPY: **Update**

  You will use VMFMAC to generate a MACLIB with the new COPY file during the rebuilding process.

- Any other types: **Replacement**

  If the new replacement parts do not already have their appropriate filetype, then they have an abbreviated form of the file type concatenated with the PTF number. Use "VM/XA SP Filetypes and Abbreviations" on page 863 to find out the complete filetype and copy the file with the complete filetype:

```
copyfile filename xxxnnnnn fm = ffffffff = (olddate ■
Ready; T=n.nn/n.nn hh:mm:ss
```

  xxx is the abbreviated filetype, *nnnnn* is the PTF number, and *ffffffff* is the complete filetype.

What to Do Next

- If you do not have a service tape, go to the instructions for rebuilding the component to which you applied local service:

  CMS Chapter 14, "Rebuilding CMS after Applying Service" on page 431
  CP Chapter 15, "Rebuilding CP after Applying Service" on page 487
  DV Chapter 16, "Service to DV," "Step 3. Build the Dump Viewing Facility" on page 512
  GCS Chapter 17, "Service to GCS," "Step 3. Build a New GCS Macro Library" on page 522

- If you have a service tape, go to the instructions for receiving service for the component to which you applied local service:

  CMS Chapter 10, "Receiving Service for CMS" on page 417
  CP Chapter 11, "Receiving Service for CP" on page 423
  DV Chapter 16, "Service to DV," "Step 1. Receive Service" on page 508
  GCS Chapter 17, "Service to GCS." "Step 1. Receive Service" on page 518.

Chapter 21. Emergency Local Service Using the Patch Facility

This chapter describes step-by-step procedures for applying patches to the CMS and CP nuclei.

Introduction

Before attempting to use the patch facility, read the following sections of this book:

- Chapter 6, "VM/XA System Product Service—An Overview" on page 349
- Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361
- Chapter 8, "Files Used in PUT and COR Service" on page 371
- "HCPLDR Command" on page 590 in Appendix B, "EXEC and Command Format Summaries"
- "The Patch Facility" on page 616 in Appendix B, "EXEC and Command Format Summaries."

The patch facility is used to circumvent problems in text decks on replacement service (object-maintained) that are part of the CMS or CP nucleus. If the problem you have is in a file that is not in the nucleus, you cannot use the patch facility.

Patches to object code (TEXT files) exploit the capability of the HCPLDR module when VER, REP, and ICS statements are included in the replacement files.

The instructions in this chapter apply to CMS and CP. Whenever there are variations for different components they are noted.

Your IBM service representative will give you the specific hexadecimal codes that need to be changed.

Step 1. Receive the Patch

In the example below, assume that you have identified the object-maintained part that is causing a problem in your VM/XA SP system. You call your IBM service representative, who tells you that there is no existing fix for the problem. He tells you that, in order to circumvent the problem, you can change the object code in HCPAFF at displacement X'061C' from X'0780' to X'0700'. This example is representative of the type of fix you may get. It changes a branch to a no-op. Finally, your representative tells you that he has assigned APAR number VM23877 to this problem. If your representative does not give you an APAR number, use a local tracking number.

1. Log on to MAINT and establish the appropriate minidisk accesses:

   ```
   vmfsetup 56643089 compid (all ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   Substitute for *compid* the component you are servicing (CMS or CP).

2. Create a patch update file on the alternate LOCAL1 minidisk:

   ```
   xedit hcpaff p23877 b ■
   input ■
   :
   file ■
   ```

   Your patch update file may look like Figure 37.

   Notes:

   1. Spaces are required between slashes and asterisks in the sample patch update file. For information on patch update file control statement syntax, see the section "The Patch Facility" on page 616.
   2. The cards appear as comments in the patch update file. The $VMFPAT$ EXEC properly inserts the patch update files into the TEXT files while running the VMFBLD module. It translates the control cards into the format required by the VMFBLD module. VMFBLD no longer invokes the HCPLDR EXEC.

```
./ * * PREREQ: NONE
./ * * CO-REQ: NONE
./ * NAME HCPAFF
./ * * This is a patch.  It
./ * *   will be replaced by
./ * *   APAR VM23877.
./ * VER 61C 0780
./ * REP 61C 0700 ■
```

Figure 37. A Sample Patch Update File

Step 2. Apply the Patch

1. Use XEDIT to create an AUX file entry on the AUXPAT level.

   There are three fields that make up an AUX entry. The first field contains the filetype of the update file. Imbedded in this filetype is the 5-digit numeric portion of the APAR number of the fix. When you originate an update, this 5-digit number should be a local tracking number. The second field is the level of the fix. For patch service you **must** use TX$ in this field. Since there is no PTF number for the patch, the third field is the APAR number or a local tracking number.

```
xedit hcpaff auxpat b ■
input ■
⋮
file ■
```

Your patch AUX file might look like Figure 37 on page 556.

```
P23877 TX$ VM23877 * A PATCH TO BE REPLACED BY APAR VM23877
```

Figure 38. A Sample Patch AUX File

   Note: You would not normally have any service on the AUXLCL level for the part for which you are creating a patch, because service on the AUXLCL level is accomplished with source update files. If you have the source to update, a patch is not necessary. If, however, you do have service at the AUXLCL level, you must rework it so that it does not conflict with the patch you are applying. You must also rework any service you have on the AUXPAT level so that it does not conflict with the patch you are applying.

2. Check the control file you are using to make sure that it includes an AUX file identification record for the AUXPAT level of service.

   "Main Control Files" on page 364 shows the contents of the default CNTRL files. The default CNTRL files for the VM/XA System Product system include the AUXPAT level at the highest level of service. If you modify the CNTRL file structure, you must make sure that the AUX file that you create for this patch is in your modified structure.

What to Do Next

Go to the instructions for rebuilding the component to which you applied the patch:

CMS Chapter 14, "Rebuilding CMS after Applying Service" on page 431
CP Chapter 15, "Rebuilding CP after Applying Service" on page 487
DV Chapter 16, "Service to DV," "Step 3. Build the Dump Viewing Facility" on page 512
GCS Chapter 17, "Service to GCS," "Step 3. Build a New GCS Macro Library" on page 522.

Chapter 22. Removing Service from VM/XA SP

This chapter describes how to remove PUT service, COR service, or local service from any component of VM/XA SP.

If you have a problem with a PTF, or if a PTF arrives for which you have applied a local circumvention, you will need to remove service you have applied.

When you install PUT service, COR service, or local service to VM/XA System Product, you build your new system on different minidisks from your old system. You should still have your old system available so you can return to it simply by accessing the old disks. In this way, of course, you remove an entire level of service. If you wish to delete only certain PTFs from VM/XA System Product, follow the procedures in this chapter.

First you must understand that many parts of a component, possibly of different kinds, may be serviced by a single PTF. You can tell what kind of part an update affects by the part's filetype:

Filetype	**Part**
ASSEMBLE	ASSEMBLE file
TEXT or TXT*nnnnn*	Text deck ("TEXT file")
EXEC or EXC*nnnnn*	EXEC
MACRO or MAC*nnnnn*	Macro
COPY or CPY*nnnnn*	Control block ("COPY file")
XEDIT or XED*nnnnn*	XEDIT macro
Note: *nnnnn* is the number of the most recently applied PTF.	

The examples in this chapter are written to assist you in understanding what has to be done to remove service from VM/XA SP. They are not meant to cover all possible situations.

1. Make a list of all the PTFs you want to remove, leaving space to write in the corresponding updates. If you know the number of the APAR you want to remove, but not the corresponding PTF number, see Appendix I, "How to Find the PTF Number from the APAR Number" on page 867.

   In this example, we suppose that you want to remove only one PTF, UM00001, from CP.

2. With every PTF, you receive a parts list, which identifies all the parts affected by that PTF. Examine the parts list:

   ```
   vmfsetup 56643089 cp (all ■
   Ready; T=n.nn/n.nn hh:mm:ss
   ```

   ```
   xedit um00001 $ptfpart fm ■
   ```

   The filename of the parts list is the PTF number; the filemode is $PTFPART. In this example, UM00001 is applied from a corrective service tape and is therefore on the CP alternate LOCAL1 disk. If it had been applied from a PUT tape, it would be on the CP alternate DELTA disk.

```
HCPABC A34567HP
HCPXYZ A34567HP
HCPABC TXT00001
```

UM00001 $PTFPART lists one update for HCPABC and one update for HCPXYZ, along with the HCPABC text deck for this PTF.

3. Since a text deck is shipped for HCPABC, it must be an ASSEMBLE file. To find out what kind of part HCPXYZ is, check its filetype:

```
listfile hcpxyz * * ■
HCPXYZ   AUXCOR   fm
HCPXYZ   MACRO    fm
HCPXYZ   MAC00004 fm
HCPXYZ   MAC00001 fm
```

HCPXYZ is a macro. (Replacement macro libraries are not shipped on IBM corrective service tapes. Macros are pulled together into a macro library when service is applied with the VMFMAC command.)

4. Comment out or delete the update entries in the auxiliary control file at the service level affected by the PTF. (It is better to comment them out, so that you can see what you did.) If the update you are commenting out or deleting is **not** at the top of the auxiliary control file, then you must also comment out or delete any updates above the update you are removing. You must also add them to your list of updates to be removed if they are not already there.

```
xedit hcpabc auxcor ■
```

Since PTF UM00001 was applied as corrective service, it should be listed in the auxiliary control file for the corrective service level. If you are using the main control file supplied by IBM, the filetype of that auxiliary control file is AUXCOR.

```
*A88888HP COR UM00003
*A56789HP COR UM00002
*A34567HP COR UM00001
 A12345HP COR UM00000
```

The update for PTF UM00001 is listed on the third line from the top.

To comment out an entry, put an asterisk in front of it. Remember to comment out the entries **above** the one you want to remove too.

```
file ■
```

```
xedit hcpxyz auxcor ■
*A99999HP COR UM00004
*A34567HP COR UM00001
file ■
```

5. When you edited HCPABC AUXCOR and HCPXYZ AUXCOR, you found three more updates that must be removed. Add these updates, along with the corresponding PTF number in the third word of the AUX entry, to your list of fixes to delete. Your list of updates that must be deleted now contains:

```
HCPABC A34567HP   PTF UM00001
HCPXYZ A34567HP   PTF UM00001
HCPABC A88888HP   PTF UM00003
HCPABC A56789HP   PTF UM00002
HCPXYZ A99999HP   PTF UM00004
```

In addition to removing the updates on this list, you must also review the parts lists for the corresponding PTFs (UM00003, UM00002, and UM00004) to find all the additional updates in those PTFs. You must remove the whole PTF, not just part of it. Save that task for later. For now, just note that you will have to look at these additional PTFs after you finish removing the one you set out to remove.

6. If the PTF you are removing does not affect any text decks, go to Step 7 on page 562. If it does, as in this example, determine whether the text deck for which you are removing service is in the load list for this component (CPLOAD EXEC, CMSLOAD EXEC, or GCSLOAD EXEC):

```
xedit cpload EXEC ■
locate /hcpabc ■
qquit ■
```

   - If the text deck for which you are removing service is in the load list, then, when you apply the next PUT tape, the correct text deck, including only the updates that you did not comment out in the HCPABC AUXCOR file, is automatically used. Continue with Step 7 on page 562.
   - If the text deck for which you are removing service is not in the load list, you must determine the text deck (TXT*nnnnn*, where *nnnnn* is the PTF number) that replaces the current HCPABC TXT*nnnnn* (which contains all four fixes, three of which you want to remove).

     a. First list all text decks for the module hit by the updates you want to remove:

```
filelist hcpabc txt* * ■

HCPABC   TXT00000 fm
HCPABC   TXT00001 fm
HCPABC   TXT00002 fm
HCPABC   TXT00003 fm
```

     b. Next, find the text deck that contains only the updates you want to keep, in this case HCPABC A12345HP. It should be the text deck corresponding to the first update you did not comment out of the auxiliary control file in Step 4 on page 560 (in this case, HCPABC TXT00000); but, to make sure, you can examine the text deck corresponding to the update at the top of the auxiliary control file (in this case, HCPABC TXT00003).

```
xedit hcpabc txt00003 ■

* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A12345HP  COR  UM00000  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A34567HP  COR  UM00001  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A56789HP  COR  UM00002  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
* DEPEND: A01010HP
A88888HP  COR  UM00003  * Comments
⋮
(Executable text is here.)
```

HCPABC TXT00003 contains the three updates you want to delete and the one update you want to keep.

Find the text deck line that contains the latest update you want to keep. This is the update immediately above the first one you want to remove. The third word of that line is the PTF

number for this update: UM00000. Prefix TXT to the numeric part of this PTF number for the filetype of the new text deck: TXT00000.

c. Since the current text deck for HCPABC contains the fixes you want to delete, you must replace it with the text deck that contains only update A12345HP. Enter:

```
copyfile hcpabc txt00000 fm hcpabc text = (replace ■
Ready; T=n.nn/n.nn hh:mm:ss
```

fm is the filemode of the alternate LOCAL1 disk.

7. Next, check each update file that you deleted or commented out of the auxiliary control file for DEPEND entries. The DEPEND entries identify previous updates that are dependent on the update you are removing. Such updates must also be removed.

   In this example, assume that only HCPABC A88888HP contains a DEPEND entry.

```
xedit HCPABC A88888HP ■
./* PREREQ: NONE
./* CO-REQ: NONE
./* IF-REQ: NONE
./* DEPEND: A01010HP
qquit ■
```

HCPABC A88888HP contains one DEPEND entry, for HCPABC A01010HP. (The filename of the dependent update file is the same as the filename of the update file you are looking at. The filetype is listed in the DEPEND entry.)

8. Add any updates you find in DEPEND entries to your list of updates that you must remove. You know the APAR number for these updates (in this example, 01010) from the five numeric characters in the filetype; but you do not yet know the corresponding PTF number. (Appendix I, "How to Find the PTF Number from the APAR Number" on page 867 will show you how to find it.) For now, put question marks in your delete list beside the DEPEND updates and come back to this point later.

   The list of updates to be deleted now looks like this:

```
HCPABC A34567HP   PTF UM00001
HCPXYZ A34567HP   PTF UM00001
HCPABC A88888HP   PTF UM00003
HCPABC A56789HP   PTF UM00002
HCPXYZ A99999HP   PTF UM00004
HCPABC A01010HP   PTF ???????
```

9. Rename the updates you want to remove with a filemode of 1:

```
rename HCPABC A34567HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
rename HCPXYZ A34567HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
rename HCPABC A88888HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
rename HCPABC A56789HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
rename HCPXYZ A99999HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
rename HCPXYZ A01010HP fm5 = = fm1 ■
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Repeat Steps 2 on page 559 through 9 for each update in your delete list. When you have done this, check the parts lists for the PTFs you noted in Step 5 on page 560 and delete any updates included in those PTFs that you missed. Then determine the PTF numbers for the DEPEND entries you found in

Step 7 (see Appendix I, "How to Find the PTF Number from the APAR Number" on page 867) and delete those PTFs.

11. Finally, rework local service to modules for which you have deleted service. (Local service is service listed at any AUX levels above the AUXCOR level in the control file for this component. See Chapter 20, "Receiving and Applying Local Service" on page 543.)

What to Do Next

- If you do not have a service tape, go to the instructions for rebuilding the component from which you removed service:

  CMS Chapter 14, "Rebuilding CMS after Applying Service" on page 431
  CP Chapter 15, "Rebuilding CP after Applying Service" on page 487
  DV Chapter 16, "Service to DV," "Step 3. Build the Dump Viewing Facility" on page 512
  GCS Chapter 17, "Service to GCS," "Step 3. Build a New GCS Macro Library" on page 522

- If you do have a service tape, go to the instructions for receiving service for the component from which you removed service:

  CMS Chapter 10, "Receiving Service for CMS" on page 417
  CP Chapter 11, "Receiving Service for CP" on page 423
  DV Chapter 16, "Service to DV," "Step 1. Receive Service" on page 508
  GCS Chapter 17, "Service to GCS," "Step 1. Receive Service" on page 518.

Part 3. Appendixes

This book contains the following appendixes:

- Appendix A, "VM/XA System Product Regeneration Requirements" on page 567, shows the modules you must regenerate when you apply a fix.
- Appendix B, "EXEC and Command Format Summaries" on page 575, shows the format and use of EXECs and commands you may need for installing and servicing VM/XA System Product.
- Appendix C, "VM/XA System Product Starter System Information" on page 725, gives sample system definition files and other information about the VM/XA SP starter system.
- Appendix D, "Listing CP Data Areas and Control Blocks" on page 853, describes a procedure for listing VM/XA System Product data areas and control blocks.
- Appendix E, "Example of Alternate GCS Nucleus Placement" on page 855, shows how to save your GCS nucleus at a different location than the one in the sample files.
- Appendix F, "Restricted Logon Passwords" on page 859, contains the IBM-supplied list of restricted passwords.
- Appendix G, "Controlling Disk String Merges" on page 861, tells you how to prevent the automatic merging of a disk string and how to merge disk strings manually.
- Appendix H, "Service Reference Tables" on page 863, lists the standard 3-character abbreviations for the various filetypes used in VM/XA SP and the parts supplied for service.
- Appendix I, "How to Find the PTF Number from the APAR Number" on page 867, shows the procedure for determining the number of a program temporary fix from the associated APAR number.
- Appendix J, "Messages" on page 869, contains messages issued by the service EXECs.

Appendix A. VM/XA System Product Regeneration Requirements

CMS Regeneration Requirements

Whenever you apply a fix to CMS source code, you must regenerate the CMS nucleus or some CMS modules. The following table shows which you must regenerate in each case. (If a source name does not appear in the table, either the file is contained within the CMS nucleus, or it is loaded by another file [for example, DMSBTB loads DMSBTP].)

If you must regenerate the CMS nucleus, see Chapter 14, "Rebuilding CMS after Applying Service" on page 431.

Table 15 (Page 1 of 6). CMS Regeneration Requirements

Change in Source	Requires Regeneration of Module/Nucleus	EXEC Procedure to Use
DMSACC	Nucleus	VMFBLD
DMSACF	Nucleus	VMFBLD
DMSACM	Nucleus	VMFBLD
DMSAMS	AMSERV	CMSGEND
DMSASD	ASSEMBLE	CMSGEND
DMSASM	ASSEMBLE	CMSGEND
DMSASN	ASSGN	CMSGEND
DMSBAB	CMSDOS	DOSGEN
DMSBCT	DMSDFT	CMSGEND
DMSBLG	DMSDFT	CMSGEND
DMSBOF	DMSCUT	CMSGEND
DMSBOP	CMSDOS	DOSGEN
DMSBTB	CMSBATCH	CMSGEND
DMSBUS	DMSCUT	CMSGEND
DMSCCM	DMSCUT	CMSGEND
DMSCDI	DMSDFT	CMSGEND
DMSCCK	CATCHECK	CMSGEND
DMSCIA	DMSCUT	CMSGEND
DMSCLS	CMSDOS	DOSGEN
DMSCMP	COMPARE	CMSGEND
DMSCPY	Nucleus	VMFBLD
DMSCVH	CMSDOS	DOSGEN
DMSDAS	CMSDOS	DOSGEN

Table 15 (Page 2 of 6). CMS Regeneration Requirements		
Change in Source	**Requires Regeneration of Module/Nucleus**	**EXEC Procedure to Use**
DMSDFT	DMSDFT	CMSGEND
DMSDLK	DOSLKED	CMSGEND
DMSDMP	CMSDOS	DOSGEN
DMSDOS	CMSDOS	DOSGEN
DMSDSK	DISK	CMSGEND
DMSDSL	DOSLIB	CMSGEND
DMSDSV	DSERV	CMSGEND
DMSEDC	EDIT (see Note 1 on page 572)	CMSGEND
DMSEDF	EDIT (see Note 1 on page 572)	CMSGEND
DMSEDI	EDIT (see Note 1 on page 572)	CMSGEND
DMSEDX	EDIT (see Note 1 on page 572)	CMSGEND
DMSEND	CMSAMS	VSAMGEN
DMSEXG	DCSSGEN	CMSGEND
DMSEXM	EXECMAP	CMSGEND
DMSFCH	CMSDOS	DOSGEN
DMSFOR	FORMAT	CMSGEND
DMSGIO	EDIT (see Note 1 on page 572)	CMSGEND
DMSGLB	GLOBAL	CMSGEND
DMSGLO	Nucleus	VMFBLD
DMSGMF	CMSDOS	DOSGEN
DMSGND	GENDIRT	CMSGEND
DMSGTM	CMSDOS	DOSGEN
DMSGVE	CMSDOS	DOSGEN
DMSHLB	HELPCONV	CMSGEND
DMSHLD	HELPCONV	CMSGEND
DMSHLI	HELPCONV	CMSGEND
DMSHLP	HELPCONV	CMSGEND
DMSHLS	HELPCONV	CMSGEND
DMSICP	IOCP	CMSGEND
DMSIDE	Nucleus	VMFBLD
DMSICT	DMSCUT	CMSGEND
DMSIMA	IMAGEMOD	CMSGEND
DMSLAB	CMSDOS	DOSGEN
DMSLBD	LABELDEF	CMSGEND

Table 15 (Page 3 of 6). CMS Regeneration Requirements		
Change in Source	**Requires Regeneration of Module/Nucleus**	**EXEC Procedure to Use**
DMSLBM	MACLIB	CMSGEND
DMSLBT	TXTLIB	CMSGEND
DMSLCK	CMSDOS	DOSGEN
DMSLDF	CMSDOS	DOSGEN
DMSLDS	LISTDS	CMSGEND
DMSLIC	CMSDOS	DOSGEN
DMSLLU	LISTIO	CMSGEND
DMSLMX	TAPE (see Note 2 on page 572)	CMSGEND
DMSMCM	CMSDOS	DOSGEN
DMSMDP	MODMAP	CMSGEND
DMSMES[*y*]	DMSMES[*y*] (see Note 3 on page 572)	VMFNLS
DMSMGC	GENMSG	CMSGEND
DMSMGD	GENMSG	CMSGEND
DMSMGE	GENMSG	CMSGEND
DMSMVE	MOVEFILE	CMSGEND
DMSOPL	CMSDOS	DOSGEN
DMSOR1	CMSDOS	DOSGEN
DMSOR2	CMSDOS	DOSGEN
DMSOR3	CMSDOS	DOSGEN
DMSNXD	NUCXDROP	CMSGEND
DMSOPT	OPTION	CMSGEND
DMSOSR	OSRUN	CMSGEND
DMSOVR	SVCTRACE	CMSGEND
DMSOVS	DMSOVS	CMSGEND
DMSPIO	Nucleus	VMFBLD
DMSPBS	DMSCUT	CMSGEND
DMSPCA	DMSPCC	CMSGEND
DMSPCB	DMSPCC	CMSGEND
DMSPCC	DMSPCC	CMSGEND
DMSPCR	DMSPCC	CMSGEND
DMSPCT	DMSPCC	CMSGEND
DMSPCW	DMSPCC	CMSGEND
DMSPOA	PROPLIB (see Note 4 on page 572)	CMSGEND

Table 15 (Page 4 of 6). CMS Regeneration Requirements

Change in Source	Requires Regeneration of Module/Nucleus	EXEC Procedure to Use
DMSPOC	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOD	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOE	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOL	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPON	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOP	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOQ	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOR	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPOS	PROPLIB (see Note 4 on page 572)	CMSGEND
DMSPRE	PRELOAD	CMSGEND
DMSPRT	Nucleus	VMFBLD
DMSPRV	PSERV	CMSGEND
DMSPUN	PUNCH	CMSGEND
DMSRDC	READCARD	CMSGEND
DMSRDR	RDR	CMSGEND
DMSRNE	RENUM	CMSGEND
DMSRPG	CMSDOS	DOSGEN
DMSRRV	RSERV	CMSGEND
DMSRSV	RESERVE	CMSGEND
DMSSAP	DMSCUT	CMSGEND
DMSSCR	EDIT (see Note 1 on page 572)	CMSGEND
DMSSFD	SAVEFD	CMSGEND
DMSSMG	DMSDFT	CMSGEND
DMSSNC	DMSCUT	CMSGEND
DMSSPA[*y*]	DMSSPA[*y*] (see Note 3 on page 572)	VMFNLS
DMSSPR	SETPRT	CMSGEND
DMSSRT	SORT	CMSGEND
DMSSSK	SETKEY	CMSGEND

Table 15 (Page 5 of 6). CMS Regeneration Requirements		
Change in Source	**Requires Regeneration of Module/Nucleus**	**EXEC Procedure to Use**
DMSSTX	CMSDOS	DOSGEN
DMSSUB	CMSDOS	DOSGEN
DMSSUP	DMSCUT	CMSGEND
DMSSYN	SYNONYM	CMSGEND
DMSSVL	CMSDOS	DOSGEN
DMSTMA	TAPEMAC	CMSGEND
DMSTPD	TAPPDS	CMSGEND
DMSTPE	TAPE (see Note 2 on page 572)	CMSGEND
DMSTPF	TAPE (see Note 2 on page 572)	CMSGEND
DMSTPG	TAPE (see Note 2 on page 572)	CMSGEND
DMSTPH	TAPE (see Note 2 on page 572)	CMSGEND
DMSTPI	TAPE (see Note 2 on page 572)	CMSGEND
DMSTPJ	TAPE (see Note 2 on page 572)	CMSGEND
DMSTRC	DMSCUT	CMSGEND
DMSTRT[*y*]	DMSTRT[*y*] (see Note 3 on page 572)	VMFNLS
DMSTYP	TYPE	CMSGEND
DMSUPD	UPDATE	CMSGEND
DMSUSR	DMSDFT	CMSGEND
DMSUTL	LOADLIB	CMSGEND
DMSVAN	CMSAMS	VSAMGEN
DMSVAS	CMSAMS	VSAMGEN
DMSVAX	CMSAMS	VSAMGEN
DMSVIP	CMSVSAM	VSAMGEN
DMSVIS	CMSDOS	DOSGEN
DMSVLT	CMSDOS	DOSGEN
DMSVVN	CMSAMS	VSAMGEN
DMSVVS	CMSAMS	VSAMGEN
DMSXCP	CMSDOS	DOSGEN
DMSXMS	DMSXMS (see Note 5 on page 572)	CMSGEND
DMSXRE	DMSXRE (see Note 5 on page 572)	CMSGEND
DMSZAP	ZAP	CMSGEND
DMSZIT	EDIT (see Note 1 on page 572)	CMSGEND

Table 15 (Page 6 of 6). CMS Regeneration Requirements

Change in Source	Requires Regeneration of Module/Nucleus	EXEC Procedure to Use
IOPC*xxxx*	IOCP	CMSGEND
IOPP*xxxx*	IOCP	CMSGEND
VMFCLEAR	VMFCLEAR	GENMOD
VMFDATE	VMFDATE	CMSGEND
VMFDOS	VMFDOS	CMSGEND
VMFLOAD	VMFLOAD	CMSGEND

Notes:

1. When invoked for EDIT, the CMSGEND EXEC procedure creates the EDIT module and then reinvokes itself to create the EDMAIN module.
2. When the CMSGEND EXEC is invoked for TAPE, it creates the TAPE module and then reinvokes itself to create the DMSLMX and DMSTP*x* modules.
3. *y* is the country code for your system national language. See Table 5 on page 311.
4. You receive messages DMSSLK0008W and DMSSOP036E when you regenerate PROPLIB. You can ignore them.
5. All EDIT source files, except DMSXMS and DMSXRE, are contained within the CMS nucleus.

CP Regeneration Requirements

If you apply corrective service to certain CP source files, you must also regenerate the corresponding CMS modules and stand-alone utilities that can be IPLed. The following table shows the source name, module name, and procedures used for regenerating the CMS module.

The UTILITY EXEC is described in Appendix B, "EXEC and Command Format Summaries" on page 575.

Table 16. CP Regeneration Requirements

Change in Source	Requires Regeneration of Module/Nucleus	EXEC Procedures to Use
HCPBSL	3CARD LOADER	UTILITY
HCPCCF, HCPFAA-H, HCPFAL-M, HCPFAR, HCPFON	CPFMTXA	UTILITY
HCPCCU	HCPCCU	UTILITY
HCPDDC, HCPDDR, HCPDDT, HCPDNC, HCPDNT	IPL DDRXA	UTILITY
HCPDIR	DIRECTXA	UTILITY
HCPED*x*	DUMPLOAD	UTILITY
HCPIFC, HCPREA	CPEREPXA	UTILITY
HCPIMG	GENIMAGE	UTILITY
HCPLDR	HCPLDR	UTILITY
HCPLDL, HCPMDLAT	CPLOAD EXEC	UTILITY
HCPMES[*y*]	HCPMES[*y*] (see Note below)	VMFNLS
HCPNMT	IMAGELIB	UTILITY
HCPOPTNS	HCPXA, HCPLDRCM	VMFMAC
HCPOVE	OVERRIDE	UTILITY
HCPRET	RETRIEVE	UTILITY
HCPSADMP	HCPSADMP	UTILITY

Note: *y* is the country code for your system national language. See Table 5 on page 311.

Dump Viewing Facility Regeneration Requirements

If you apply corrective service to the dump viewing facility TEXT files, you must also regenerate the dump viewing facility. The following table shows the TEXT filename, and the procedure for regenerating the dump viewing facility.

Table 17. Dump Viewing Facility Regeneration Requirements

Change in TEXT file	Requires Regeneration of Module	EXEC Procedures to Use
HCS*xxx*	ADDMAP, DUMPSCAN, MAP, PRTDUMP, SCAN, TRACERED	UTILITY

GCS Regeneration Requirements

If you have local modifications to the GCS message repository, you must regenerate it. The following table shows the filename and the procedure to be used for regenerating the dump viewing facility.

Table 18. GCS Regeneration Requirements

Change in Source	Requires Regeneration of Module	EXEC Procedures to Use
CSIMES[*y*]	CSIMES[*y*]	VMFNLS
Note: *y* is the country code for your system national language. See Table 5 on page 311.		

Appendix B. EXEC and Command Format Summaries

This section is a general reference for commands, EXECs, and MODULEs you may use during system generation or service application. The commands and EXECs that appear here are:

- ASMGEND EXEC
- CMSGEND EXEC
- DCSSGEN command
- DIRECTXA command
- DISKMAP EXEC
- DOSGEN EXEC
- GROUP EXEC
- HCPLDR command
- INSTFPP EXEC
- ITASK EXEC
- The Patch Facility
- PRELOAD MODULE
- SAMGEN EXEC
- SAMPNSS EXEC
- SPLOAD EXEC
- UPDATE command
- UTILITY EXEC
- VMFAPPLY EXEC
- VMFBLD EXEC
- VMFHASM EXEC
- VMFLKED EXEC
- VMFMAC EXEC
- VMFMERGE EXEC
- VMFNLS EXEC
- VMFOVER EXEC
- VMFPLC EXEC
- VMFPLCD EXEC
- VMFPLC2 command
- VMFREC EXEC
- VMFREMOV EXEC
- VMFSETUP EXEC
- VMFVIEW EXEC
- VMFZAP EXEC
- VSEVSAM EXEC
- ZAP MODULE
- ZAPTEXT EXEC.

ASMGEND EXEC

The ASMGEND EXEC procedure builds the system assembler and creates the associated auxiliary directory. ASMGEND loads the text decks for the assembler in the correct overlay structure and produces a load map.

Format

The format of the ASMGEND EXEC is:

ASMGEND	

Usage Notes

1. The assembler text decks normally reside on the system S-disk in filemode S1. This disk must be accessed in some additional filemode before issuing this command in order to locate these files. For example:

   ```
   access 190 a ■
   access 193 b ■
   ```

2. Use the ASMGEND EXEC if you have modified the assembler (IFOX*nn* MODULE) source. If you have not modified this source, and wish to create the assemble module, possibly after modifying DMSNGP or after creating a new CMS system disk, use the CMSGEND EXEC.

Messages

ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED.

Explanation: Enter the mode letter of the disk containing the assembler modules. The ASSEMBLE command accesses this disk during processing. If you enter a mode letter, ASMGEND uses that mode letter as the *targetmode* operand of the GENDIRT command when it creates the auxiliary directory. If you do not specify a mode letter, S is used.

ASSEMBLE XF GEND COMPLETE.

Explanation: The system assembler and its associated auxiliary directory have been generated successfully.

ASSEMBLE XF GEND FAILED.

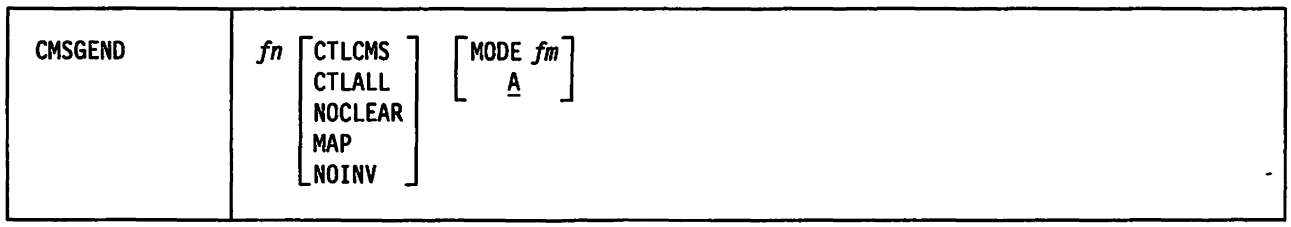
Explanation: The system assembler text files were not loaded successfully.

CMSGEND EXEC

Use the CMSGEND EXEC procedure to generate a new CMS module or LOADLIB from a text file and place the new CMS module or LOADLIB on the specified disk.

Format

The format of the CMSGEND EXEC is:



Where:

fn
is the filename of the CMS module or LOADLIB that is to be generated by the CMSGEND EXEC. Only one filename may be specified in the CMSGEND command line.

The filenames that may be specified in the CMSGEND command are any disk-resident CMS commands and service programs.

CTLCMS
displays each CMS command as it is executed in the CMSGEND EXEC procedure. This is equivalent to the EXEC statement &CONTROL CMS.

CTLALL
displays every executable statement as it is executed in the CMSGEND EXEC procedure. This is equivalent to the EXEC statement &CONTROL ALL.

NOCLEAR
specifies that the CLEAR option is not to be issued when CMSGEND invokes the LOAD command.

MAP
specifies that the NOMAP option is not to be issued when CMSGEND invokes the GENMOD command.

NOINV
issues the NOINV option when CMSGEND invokes the LOAD command; this suppresses the displaying of invalid cards at the terminal. If the text deck was created with the VMFASM EXEC, it may contain update listing information; these records are displayed during the loading process unless you specify NOINV.

MODE *fm*
A
indicates the access mode of the disk to receive the new module. Filemode A is the default.

Usage Notes

1. The assembler text decks normally reside on the system S-disk in filemode S1. This disk must be accessed in some additional filemode before you issue this command in order to locate these files. For example:

   ```
   access 190 a ■ (CMS system disk)
   access 293 b ■ (CMS DELTA disk)
   access 193 c ■ (CMS BASE disk)
   ```

2. Use the CMSGEND EXEC if you have not modified the assembler (IFOX*nn*) source, and wish to create the assemble module, possibly after modifying DMSASM, DMSASN, DMSASD, or creating a new CMS system disk.
3. You can also use the CMSGEND EXEC to regenerate the ASSEMBLE command when you move the CMS system disk. When you specify ASSEMBLE, CMSGEND prompts you to enter a disk mode letter so it can refresh the assembler's auxiliary directory. (Use the ASMGEND EXEC procedure if you are updating the assembler.)
4. When using CMSGEND EXEC to regenerate the PROP command, you are only generating the PROPLIB LOADLIB. (The CMSGEND options NOCLEAR, MAP, and NOINV have no effect when generating the PROP command.)
5. The filetype of the text deck must be TEXT.

How CMSGEND Works

CMSGEND keeps a list of the CMS disk-resident modules and LOADLIBs, the filenames of the text files used to create them, and any special attributes required to generate them. For example, the PRINT command must be generated with the ORIGIN TRANS and the SYSTEM options. It is composed of the DMSPRT text file. To generate a new PRINT module, issue:

```
cmsgend print ■

*** CURRENT STATUS:
FILE ' PRINT MODULE A2 ' DOES NOT EXIST
FILE ' PRINT MODULE A1 ' DOES NOT EXIST

*** LOADING:
 INVALID CARD - *  DMSGPI  MACLIB  A1 5.6CMS mm/dd/yy hh:mm

 :
 DMSPRT     SO 00E000
 PRINT         00E000

 ***RESULTS:
 ' PRINT MODULE A2 ' CREATED FROM TEXT DECK ( S ) DMSPRT

 WITH OPTIONS TRAMX SYSTEM NOMAP

 R;
```

Messages

***** CURRENT STATUS: "*fn* MODULE A*n*" [EXISTS|DOES NOT EXIST].**

Explanation: This message indicates whether a generated module already exists.

***** LINK EDITING: *fn* TEXT.**

Explanation: This message indicates that CMSGEND is link editing *fn* TEXT to create a LOADLIB. The existing LOADLIB is erased and not renamed when generating a new one.

***** LOADING:**

Explanation: This message indicates that CMSGEND is loading the text decks.

***** (UNDEF. NAMES NORMAL FOR EDMAIN)**
***** NOW WE HAVE A SECOND PASS FOR EDMAIN MODULE.**

Explanation: These messages indicate that the EDIT command requires two passes to resolve undefined names.

***** NOW WE HAVE A SECOND PASS FOR DMSTPI MODULE.**

Explanation: This messages indicates that the TAPE command requires two passes to generate the necessary modules. The first pass generates the DMSLMX module as the TAPE module; the second pass generates the DMSTPE, DMSTPG, DMSTPH, DMSTPI, and DMSTPJ text files as the DMSTPI module.

***** RESULTS:**
['*fn*' MODOLD A1' WAS ERASED]
['*fn* MODULE A2' RENAMED TO '*fn* MODOLD A1']

'*fn* MODULE A2' CREATED FROM TEXT DECK(S) ...
WITH OPTIONS ...

Explanation: These messages indicate which existing modules were erased and renamed, which text files were used to create the new module, and the attributes used to create the module.

ENTER GENDIRT TARGET DISK MODE LETTER
(NULL LINE DEFAULTS TO "S" DISK).

Explanation: This message is issued when you specify ASSEMBLE. You should enter the mode letter of the disk that contains the system assembler. This letter is used as the target disk mode address for the GENDIRT command.

***** ERROR MESSAGE ISSUED IS NORMAL FOR LINK EDITING.**

Explanation: If the TEXT deck was created with VMFASM EXEC, it may contain update listing information; these records will cause the linkage editor to generate an error message. The error is normal.

ERROR OCCURRED. CMSGEND STOPS.

Explanation: This message indicates that an error occurred and that CMSGEND processing ends.

INVALID ARGUMENT *fn*.

Explanation: This message indicates an invalid filename was specified on the command line.

TYPE 'CMSGEND *fn* <options>'.

Explanation: This message indicates that no filename was specified on the command line.

DCSSGEN Command

To install CMSINST, use the DCSSGEN command procedure. This command lets you build, load, and save a CMSINST segment that contains the EXECs and Editor macros that you select for your installation.

The installation segment (CMSINST is the default name) is designed to contain EXECs and System Product Editor (XEDIT) macros. When a frequently used EXEC or Editor macro resides in a saved segment, multiple users can share the same executing copy.

Before you issue the DCSSGEN command you must create a file that contains a load list of the EXECs and System Product Editor (XEDIT) macro instructions to be loaded into the segment. Information about the load list follows this description of the DCSSGEN command.

Format

The format of the DCSSGEN command is:

DCSSGEN	*filename filetype filemode* [*segname* / CMSINST]

Where:

filename filetype filemode
is the file ID of your load list.

segname
CMSINST
is the name that you want to assign to the saved segment (*segname*). The default segment name is CMSINST.

Note: The values for this command are positional.

DCSSGEN performs the following operations:

- Processes the loadlist file, sequentially loading each EXEC and Editor macro into storage
- Saves the segment
- Writes a load map to the A-disk as *segname* DCSSMAP A.

The Load List for the DCSSGEN Command

The loadlist must be a fixed-format file with a logical record length of 80. Each record in the file must contain the fileid of one EXEC or System Product Editor macro or a comment. DCSSGEN processes the records sequentially.

The format of a DCSSGEN loadlist entry is:

fn ft [*fm* [*execname* [*exectype*]]]

Where:

fn
is the filename of the EXEC or editor macro to be loaded.

ft
is the filetype of the EXEC or editor macro to be loaded.

fm
is the filemode of the EXEC or editor macro to be loaded. If the filemode is specified as *, DCSSGEN loads the first file in the disk search order that satisfies the filename and filetype qualifications.

execname
is the filename to be assigned to the loaded EXEC or editor macro. The default is "=", which means that the present filename is to be used.

exectype
is the filetype to be assigned to the loaded EXEC or editor macro. The default is "=", which means that the present filetype is to be used.

The filename and filetype of the EXEC or editor macro can each be from 1 to 8 characters. The valid characters are A-Z, a-z, 0-9, $, #, @, +, - (hyphen), : (colon), and _ (underscore). The execname and exectype may also be from 1 to 8 characters. However, they are not limited to the filename and filetype character set. The only characters NOT valid within an execname and exectype are =, *,) (right parenthesis), ((left parenthesis), and X'FF'.

To enter a comment in the loadlist, type an asterisk (*) in column one followed by the text of the comment.

For example, your loadlist entries may look like this sample:

```
* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S
```

Before you process your loadlist, remove the comments and unnecessary blanks from the source program to conserve storage space. The EXECUPDT command with the NOCOMMENTS option removes all comments and leading blanks. One comment line containing the exec name and exec type is inserted at the beginning of the file. If the source file contains Double-Byte Character Set (DBCS) characters, also specify the ETMODE option. For more information about the EXECUPDT command, refer to *VM/XA SP CMS Command Reference*.

In the load map file, the records copied from your loadlist file are left-justified. The records created during the build process are indented five spaces. Comments are also copied from your loadlist file, with an asterisk (*) in column one followed by the text.

The load map file (CMSINST DCSSMAP A) for the sample loadlist would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
    15:41:59 10/22/85 copy of RDRLIST  EXEC     S loaded as MAIL     EXEC
    EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
    15:41:58 10/22/85 copy of FILELIST EXEC     S loaded as FILELIST EXEC
    EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
     7:30:18 11/26/85 copy of SYSPROF  EXEC     S loaded as SYSPROF  EXEC
    EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
     8:47:55 12/18/84 copy of PARSE    XEDIT    S loaded as PARSE    XEDIT
    EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
    15:40:32 10/22/85 copy of DISCARD  EXEC     S loaded as DISCARD  EXEC
    EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
    11:03:53 10/24/85 copy of NOTE     EXEC     S loaded as NOTE     EXEC
    EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
    15:41:55 10/22/85 copy of PROFNOTE XEDIT    S loaded as PROFNOTE XEDIT
    EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
    15:41:05 10/22/85 copy of ALL      XEDIT    S loaded as ALL      XEDIT
    EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
    *** End of Source List ***
    CMSINST built at 15:56:34 on 12/02/86
```

DIRECTXA Command

DIRECTXA is a CMS command used to create a CP user directory.

Note: This command works in a System/370 mode virtual machine only.

Format

The format of the DIRECTXA command is:

DIRECTXA	[*fn* [*ft* [*fm*]]] [(*options...*[)]] USER DIRECT * *options:* [EDIT] [MIXED]

Where:

fn
USER

is the file name of the directory. If not specified, the file name is USER.

ft
DIRECT

is the file type of the directory. If not specified, the file type is DIRECT.

fm
*

is the filemode of the directory. If not specified, the file type is * (the command searches all accessed disks for the file name and file type).

options:

EDIT
allows you to issue the DIRECTXA command without updating the directory on disk. With this option, you can check the syntax of directory control statements.

MIXED
specifies that the DIRECTXA command is to ignore VM/SP statements and options encountered in the directory. This allows you to use a directory from a VM/SP system. DIRECTXA prints informational messages for each statement or option it ignores.

How the Directory Program Works

The DIRECTXA command performs the following steps during execution:

1. The program looks for the file you specified on the DIRECTXA command line. If you did not specify a file name or a file type, the program looks for a file that has the file name of USER and a file type of DIRECT.
2. If the program does not find the directory or an error occurs during processing, the program does not create a directory and the old directory remains on line. The program will continue to check the syntax of all control statements before ending.

3. DIRECTXA looks for the RPWLIST DATA file, which contains a list of restricted passwords. (The RPWLIST DATA file supplied by IBM is printed in Appendix F, "Restricted Logon Passwords" on page 859. You can edit it to add your own restricted passwords.) If the RPWLIST DATA file is not found, DIRECTXA issues a warning message but continues processing. If the RPWLIST DATA file is found, DIRECTXA checks all the passwords in the directory against it and changes any restricted passwords (except the password of the user issuing the DIRECTXA command) to NOLOG. Any user whose password is changed to NOLOG will not be able to log on. If the password of the virtual machine issuing the DIRECTXA command is restricted, it is not changed to NOLOG. DIRECTXA issues an error message and does not update the directory.
4. If DIRECTXA runs out of DASD space, the program prints an error message and ends immediately. To continue, you must allocate more cylinders for the user directory.

   If DIRECTXA runs out of virtual storage, the command issues an error message and ends immediately. To continue, you must define more virtual storage (DEFINE STORAGE command). Using the following formula, calculate the size of virtual storage you need:

   vstor = 50 Kb + (16 bytes × *userids*)

   Where:

   vstor
   is the amount of virtual storage in the program area that DIRECTXA requires.

   userids
   is the total number of users defined in the directory.
5. If no errors occur, the program writes the directory on DASD and writes the DASD address of the new directory on the volume label for that device. If the program is updating an active system directory, the program makes the new directory immediately available for the system.

Usage Notes

1. A complete description of the directory program, including a description of directory control statements, is contained in *VM/XA SP Planning and Administration.*
2. When writing a user directory, DIRECTXA does not overwrite the current directory, but does write an alternate user directory. The directory pointer in cylinder 0 is then changed to reference this new directory. It is advisable, therefore, to issue the DIRECTXA command twice. This assures that the directory you wish to use is available; otherwise, you may lose your directory again.

Restrictions

1. The DIRECTXA command executes only under CMS.
2. You must have CP privilege class A, B, or C in order to update the system user directory.
3. You cannot use the DIRECTXA command to create a directory in VM/XA SP Release 2 format on the VM/XA SP Release 1 system residence volume. You must create the Release 2 directory on a different physical pack.

Examples

To compile the USER DIRECT source directory and apply it to the system if successful, enter:

```
directxa user direct * ■
```

To do a test compile of a USER DIRECT source directory from a VM/SP HPO 5 system, ignoring control statements that are valid only for VM/SP or that contain options valid only for VM/SP, enter:

```
directxa user direct * (edit mixed ■
```

Messages

VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.0.

Explanation: You have invoked the directory program.

EOJ DIRECTORY UPDATED AND ON LINE.

Explanation: The directory has been updated. The new directory has been placed in use by CP.

EOJ DIRECTORY UPDATED.

Explanation: The directory has been updated, but has not been placed in use by CP. You receive this response if you write the directory to a CP-formatted DASD that is not a CP-owned volume, or if you do not have the necessary privilege class to change the CP directory that is in use.

EOJ DIRECTORY NOT UPDATED.

Explanation: The directory has not been updated. You receive this response if you issue the DIRECTXA command with the EDIT option. You also receive this response if errors prevent the directory from being updated.

Return Codes

The DIRECTXA command issues the following return codes:

Return Code	Possible Causes
0	DIRECTXA executed successfully. The real CP directory has been updated (unless the EDIT option was specified).
1	The directory source file was not found on an accessed disk.
2	An error was encountered while processing the directory source file.
3	An invalid option was specified on the DIRECTXA command line.
4	No errors were encountered, but you do not have the proper privilege class to update the real CP directory.
5	Condition code 1 was received from the DIAGNOSE X'3C'. A class A, B, or C user updated a virtual directory.
6	Condition code 2 was received from the DIAGNOSE X'3C'. An invalid directory pointer was found on cylinder 0 record 3.

Return Code	Possible Causes
7	Condition code 3 was received from the DIAGNOSE X'3C'. A fatal I/O error occurred.
9	The directory has been rewritten, but warning messages have been issued.
> 100	Return codes greater that 100 may be returned and will be accompanied by message 764. See the explanation for message 764 for details on these return codes.
333	DIRECTXA was run in EDIT mode, and at least one invalid password was changed to NOLOG.

DISKMAP EXEC

The DISKMAP EXEC summarizes the MDISK statements in the user directory. The output produced by this EXEC shows gaps and overlaps between minidisk assignments.

Format

The format of the DISKMAP EXEC is:

Where:

filename
is the filename of the directory to be mapped.

filetype
DIRECT
is the filetype of the directory to be mapped. The default is DIRECT.

The output from the DISKMAP EXEC is a file sent to your A-disk. The filename of the output is the same as that of the target directory. The filetype of the output is DISKMAP. The file contains information on MDISK statements found in the directory. The files are in order by volume in the output file. Gaps between minidisks and overlapping minidisks are flagged.

DISKMAP does **not** replace the EDIT function of the DIRECTXA command. You should use both to check your directory after changes. (For a description of the DIRECTXA command, see "DIRECTXA Command" on page 583.)

Usage Notes

1. Because some DASD types come in several sizes, DISKMAP does not list gaps found after all minidisks. You need to know the maximum cylinder/block value for your DASD type.
2. DISKMAP creates both the map and a workfile on your A-disk. If your directory is very large and your A-disk is almost filled, you may need to find some extra disk space in order to run DISKMAP.
3. You may choose to include some overlaps in the directory. DISKMAP flags *all* overlaps; you must understand your layout to determine if a particular overlap is expected or in error.

Example

To see how DISKMAP works, enter:

```
diskmap user ■
```

This produces a map of the sample directory that you loaded during installation of VM/XA SP.

DOSGEN EXEC

Use the DOSGEN EXEC to create the segment called CMSDOS, which contains text files needed to create a CMS/DOS environment that simulates DOS/VSE (Disk Operating System/Virtual Storage Extended) under CMS.

To install CMSDOS, use the DOSGEN EXEC procedure in Step 28 of Chapter 2, "Installing VM/XA System Product Release 2.1 with the Starter System (First Level)," Step 28 of Chapter 3, "Installing VM/XA System Product Release 2.1 with the Starter System (Second Level)," or Step 30 of Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System." You invoke DOSGEN with the hex load address and the name that you want to assign to this segment. This load address and name must match the address and name that you defined for this segment.

Notes:

1. The CMSDOS segment must be defined at a higher address than the CMSBAM segment.
2. Before you load and save CMSDOS, you must define your virtual machine with at least 512KB free storage above the end of the CMSDOS segment. This provides room for the loader tables, which occupy the top pages of virtual storage. After you load and save the segment, the loader tables and the 512KB free storage are no longer required.

DOSGEN performs the following operations:

- Checks that the specified virtual address contains valid characters and that it is greater than X'20000' and less than 16MB
- Looks for a read/write accessed A-disk on which to write the CMS loader work file
- Loads all the text files needed for VSE simulation, starting at the address specified
- Assigns a storage protection key of **X'D'**
- Saves the segment
- Writes the load map to the A-disk as LOAD MAP A5.

Messages

DMSGEN095E INVALID ADDRESS.

Explanation: DOSGEN detected an error in the address that you specified.

DMSGEN006E NO READ/WRITE A-DISK ACCESSED.

Explanation: DOSGEN could not find a read/write accessed A-disk.

DMSGEN111E DOSGEN FAILED DUE TO LOAD ERRORS.

Explanation: DOSGEN found unresolved external references while loading the text files.

DMSGEN412S DOSGEN FAILED DUE TO SETKEY ERRORS.

Explanation: DOSGEN detected an error while assigning the storage key.

DMSGEN141S DOSGEN FAILED DUE TO SAVESYS ERRORS.

Explanation: DOSGEN detected an error while saving the segment.

GROUP EXEC

Use the GROUP EXEC to set up a group control system (GCS) configuration file. You can invoke the GROUP EXEC from the CMS environment. When you do, you see a series of panels on which you enter information about your virtual machine group.

Format

The format of the GROUP EXEC is:

GROUP	[*systemname*]

Where:

systemname
is an optional parameter that specifies the filename that you want to assign to the configuration file. If you enter this parameter with the GROUP command, then the Primary Option Menu panel appears with the SYSTEM NAME field filled in. If you enter the GROUP command without this parameter, then you must complete the SYSTEM NAME field on the Primary Option Menu panel. In either case, the system name that you specify must match the SYSNAME entry for this GCS system in the DMKSNT file. The sample SYSNAME entry is GCS.

For an example of using the GROUP EXEC panels, see "Step 26. Load, Build, and Save GCS" on page 77.

HCPLDR Command

The HCPLDR command invokes and controls the system loader to perform one of two possible functions:

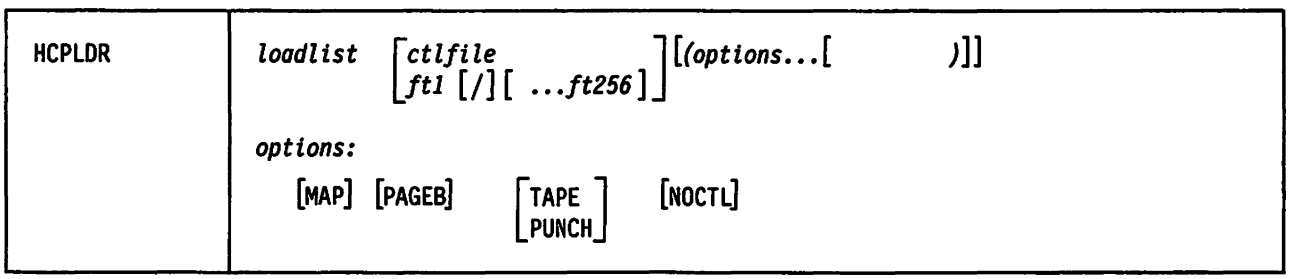
- If you issue the HCPLDR command without the PUNCH option or the TAPE option, the loader generates a system (directly under CMS); this means the system loader link-edits all object files and passes control to the program (HCPGEN) that writes the system on the system residence device.
- If you issue the HCPLDR command with either the TAPE option or the PUNCH option, the loader creates a continuous card deck beginning with a three-card loader, continuing with a stand-alone version of the system loader, and ending with the object files belonging to the system to be generated. When you issue HCPLDR with the TAPE or PUNCH option to generate CP, the loader will not link-edit object modules and will not pass control to HCPGEN to write the system on the system residence device.

Issue the HCPLDR command under CMS.

Note: This command works in a System/370 mode virtual machine only.

Format

The format of the HCPLDR command is:



Where:

loadlist
is the filename of the load list EXEC file; it contains the names of the object files in the order in which they are to be loaded. The load list may look like:

```
&CONTROL OFF
&1 &2 &3 HCPLDR LOADER
&1 &2 &3 fn [ft]
&1 &2 &3 fn [ft]
⋮
```

fn and *ft* are the filename and filetype of an object file. The *ft* is optional. If *ft* is omitted, HCPLDR searches the control file or an explicit list of filetypes to determine the filetype.

HCPLDR LOADER must always be the first file. HCPLDR loads each object module in order from the top of the load list to the bottom.

You can place most loader control statements in the load list EXEC file. You cannot place VERIFY, REPLACE, or INCLUDE CONTROL SECTION statements in the EXEC file. A loader control statement begins with a 12-2-9 punch (X'02'). Whenever HCPLDR encounters X'02' in the first column of a record, it interprets that record as a control statement. For more information see "Loader Control Statements" on page 592.

ctlfile
is the filename of the control file. For more information about control files, see "UPDATE Command" on page 631.

HCPLDR uses the update level identifiers in the control file to determine the filetype for those object modules listed in the load list without filetypes. The command skips the MACS record in the control file and searches for update level identifiers from the top of the control file down. If HCPLDR finds an update level identifier other than TEXT, the command appends the identifier to TXT to create a filetype, then searches for the file (*fn* TXT*xxxxx*). If HCPLDR cannot find an object file that has the first update level identifier, the command finds the next identifier and continues the search. As a last resort, when HCPLDR cannot find the object file according to update level identifiers, the command searches for *fn* TEXT. For example, if the control file is:

```
TEXT MACS HCPM20 HCPM1D
SPEC AUX1111
IBM1 AUXR21
```

HCPLDR follows this search order:

```
fn TXTSPEC
fn TXTIBM1
fn TEXT      (last resort)
```

If an asterisk (*) or a period appears in the first column of a control file record, HCPLDR ignores the record. The command also ignores update level identifiers of PTF.

ft1 [/] [*...ft256*]
is a list of filetypes (up to 256) that you want HCPLDR to use in lieu of a control file. HCPLDR establishes a search order from left to right. The list of filetypes can be null, in which case a filetype must be specified for each object module in the load list EXEC.

If you place a slash (/) anywhere in the filetype list, HCPLDR notifies you when it loads an object module that has a filetype to the left of the slash.

You must use the NOCTL option if you specify filetypes directly on the HCPLDR command line.

Assume you want to establish the following search order for HCPLDR:

```
fn TXT1
fn TXT2
fn TXT3
```

Also, you want to be notified when HCPLDR uses filetype TXT1 or TXT2. The load list you are using is CPLOAD EXEC. Enter the following command:

```
hcpldr cpload txt1 txt2 / txt3 (noctl ■
```

When you enter a list of filetypes on the command line, HCPLDR does not look for *fn* TEXT automatically.

Options:

MAP
requests HCPLDR to produce a load map of the system it creates. The load map is called *loadlist* MAP A1.

This option is not valid if you use the TAPE or the PUNCH option. If you use the TAPE or PUNCH option, the system loader creates a load map when you load (IPL) the tape or reader.

PAGEB
requests that every CSECT be placed on a 4-kilobyte page boundary.

This option is not valid if you use the TAPE or the PUNCH option.

TAPE
sends the output of object files to a tape drive at virtual device number 0182. At the command execution, HCPLDR does not link-edit the system; rather, HCPLDR places a 3-card loader and a stand-alone loader before the object modules. When you load (IPL) the tape, the stand-alone loader link-edits the object modules and passes control to the program (HCPGEN) that writes the system onto the system residence device. The stand-alone loader also creates a load map of the system. HCPLDR ignores MAP or PAGEB, if you specify those options along with TAPE.

PUNCH
sends a punch file to virtual punch 000D. At the command execution, HCPLDR does not link-edit the system; rather, the command places a 3-card loader and a stand-alone loader before the object modules. You can spool your punch to your virtual reader or some other (System/370 or System/370-XA) virtual machine reader. When loaded (IPL), the stand-alone loader link-edits the object modules and passes control to the program (HCPGEN) that writes the system onto the system residence device. HCPLDR ignores MAP or PAGEB, if you specify those options along with PUNCH.

NOCTL
is the option you must use when you use a list of filetypes in lieu of a control file. HCPLDR will establish a search order of filetypes according to what you enter on the command line.

Usage Notes

1. When you use the system loader as a system generator, your virtual machine must have sufficient storage to contain CMS **plus** the entire system you are generating (in the case of CP, this includes any virtual=real region). If the system is too large to generate directly, use the stand-alone version of the system loader.
2. Using the PAGEB option improves the debugging task since it eliminates needless arithmetic (the lower three hexadecimal digits of the load address is the same as the assemble address). Note, however, that PAGEB should only be used for testing purposes because the option wastes storage.
3. After the new CP nucleus resides on the system residence device, you can shut down your present system (with the SHUTDOWN command) and perform a hardware load of the new system.
4. The following response may appear when you load the CP nucleus:

   ```
   CSECT'S WITH SIZE GREATER THAN CONDITIONAL PAGE
   BOUNDARY @MAPSTRT xxxxxx HCPEMA xxxxxx
   ```

   This response is for information only and is no cause for concern unless other CSECTs are listed. The response indicates that the CSECTs are larger than the specified page size. HCPEMA contains the error message text and @MAPSTRT is the machine-readable load map. CP handles these CSECTs in a special way. If any *other* pageable modules exceed the page size, ensure that you have not added any CPB load control statements to the load file and then notify the IBM Support Center.

Loader Control Statements

The following describe loader control statements, which can be actual cards or card images. The following general rules apply to loader control statements:

1. The first column of the loader control statement must be X'02' (the 12-2-9 punch).
2. Columns 2–4 must contain the statement acronym.
3. Unless stated otherwise, operands can appear anywhere between columns 5 and 72.
4. Unless stated otherwise, operands must consist of arguments of no more than 8 characters.
5. If a control statement has more than one operand, the operands must be separated by blanks.

Printer Control Statement

The printer control statement enables or disables printing of the cross-reference listings and load maps.

Format:

1	2–4	5–72
X'02'	PRT	{ON / *rdevno* / OFF}

Where:

X'02'
must appear in column one.

PRT
is the printer control statement acronym.

{ON | *rdevno* }
enables printing of cross-reference listings and the load map. If you specify ON, the printer is at 000E. If you specify rdevno, the printer is at that device number.

OFF
disables printing.

If you issue the HCPLDR command without the PUNCH or TAPE option, this operand is ignored.

Note: Only one operand may appear on the statement.

Set Page Boundary Control Statement

The set page boundary control statement aligns the next CSECT according to the page size specified.

Format:

1	2–4	5–72
X'02'	SPB	*nnnnn*

Where:

X'02'
must appear in column 1.

SPB
is the set page boundary control statement acronym.

nnnnn
is the page size in hexadecimal. The page size may be any power of 2 from 8 to 65536. If not specified, the page size is X'1000' (decimal 4096).

Note: Only one argument may appear on the statement.

Unconditional Page Boundary Control Statement

The unconditional page boundary control statement aligns all subsequent CSECTs according to the page size specified.

Format:

1	2–4	5–72
X'02'	UPB	*nnnnn*

Where:

X'02'
must appear in column 1.

UPB
is the unconditional page boundary control statement acronym.

nnnnn
is the page size in hexadecimal. The page size may be any power of 2 from 8 to 65536. The initial value is 8. If not specified, the page size is X'1000' (decimal 4096).

Note: Only one argument may appear on the statement.

Conditional Page Boundary Control Statement

The conditional page boundary control statement conditionally aligns all subsequent CSECTs according to the page size specified. If a CSECT will not fit within the remainder of the present page, the system loader will locate the CSECT on the next page boundary.

Format:

1	2–4	5–72
X'02'	CPB	*nnnnn*

Where:

X'02'
must appear in column 1.

CPB
is the conditional page boundary control statement acronym.

nnnnn
is the page size in hexadecimal. The page size may be any power of 2 from 8 to 65536. If not specified, the page size is X'1000' (decimal 4096).

Note: Only one argument may appear on the statement.

Usage Note: The standard value for page sizes is X'1000' (decimal 4096). Module HCPMM4 uses this standard size for the CP nucleus.

Set Location Counter Control Statement

The set location counter control statement sets the next CSECT at an absolute address.

Format:

1	2–4	5–23	24–80
X'02'	SLC	*nnnnnnnn*	ignored

Where:

X'02'
must appear in column 1.

SLC
is the set location counter acronym.

nnnnnnnn
is the absolute address in hexadecimal.

Note: Only one argument may appear on the statement.

Usage Notes:

1. Module HCPSYS uses the set location counter control statement to set the size of the virtual=real area.
2. CMS uses the set location counter control statement to set the transient and user areas.

Padding Control Statement

The padding control statement fills all unspecified locations in a CSECT with a specified value.

Format:

1	2–4	5–72
X'02'	PAD	*nn*

Where:

X'02'
must appear in column 1.

PAD
is the padding control statement acronym.

nn
is the fill character in hexadecimal. The initial value is 0.

Note: Only one argument may appear on the statement.

Usage Note: Padding helps in debugging a program; if a program references unspecified areas, you can detect such references by padding with a value of X'F0' or some similar value.

Parameter Control Statement

The parameter control statement supplies a parameter string that a loaded system uses upon execution.

Format:

1	2–4	5–72
X'02'	PRM	*parmstring*

Where:

X'02'
must appear in column 1.

PRM
is the parameter control statement acronym.

parmstring
is a parameter string having a 2-byte length field followed by the parameter. When the system begins execution, general purpose register one addresses this parameter string.

Subsystem Control Statement

The subsystem control statement specifies a subsystem and the location within a virtual machine where the subsystem appears. Such a subsystem is a separately loaded system, having its own relocation, page boundaries, cross reference listings, and load maps. However, the system loader loads the subsystem as a part of the main system. Use such a "system loaded within a system" for stand-alone programs. You can load the programs as part of the CP nucleus.

Format:

1	2–4	5–72
X'02'	SYS	*name* [*loc*]

Where:

X'02'
must appear in column 1.

SYS
is the subsystem control statement acronym.

name
is the name of the subsystem.

loc
is the location where the subsystem appears in the virtual machine. If omitted, the location is 0.

Loader Termination Control Statement

The loader termination control statement terminates the current system or subsystem. Control is passed to the entry point specified on the loader termination control statement. Every subsystem control statement must have its corresponding loader termination control statement and there must be one loader termination statement for the base system.

Format:

1	2–4	5–72
X'02'	LDT	*name*

Where:

X'02'
must appear in column 1.

LDT
is the loader termination control statement acronym.

name
is the entry point name that receives control after the system nucleus is loaded into real storage. This operand is ignored for subsystems.

Replace Control Statement

The replace control statement replaces instructions and constants in virtual storage.

Format:

1	2–4	5–6	7–12	13–14	15–16	17–70	71–72	73–80
X'02'	REP		*sadd*		*esdid*	*flds*		nu

Where:

X'02'
must appear in column 1.

REP
is the replace control statement acronym.

sadd
is the hexadecimal starting address of the area that will be replaced.

esdid
is the External Symbol Identification, a number assigned by the compiler or assembler to the CSECT in which the replacement occurs.

flds
can be up to eleven 4-digit hexadecimal fields, separated by commas, each replacing one halfword. A comma cannot follow the last field.

nu
means "not used" by the loader. The field may be left blank or contain program identification.

Usage Notes:

1. The replace control statement must be in hexadecimal code.
2. The data in columns 17–70 (excluding commas) on the replace statement replaces what already was loaded into virtual storage beginning at the address specified in columns 7–12.
3. A replace statement may appear:
   - Immediately preceding the END statement of an object module, if the module does not contain relocatable data (such as address constants)
   - Immediately preceding the first relocatable dictionary statement, if the object module does contain relocatable data
4. If additions made by the replace statements increase the length of a CSECT, an include control section control statement must appear preceding the object module.
5. If the most recent VER control statement for a CSECT failed, the replace operation for that CSECT does not occur.

Verify Control Statement

The verify control statement verifies a corresponding replace control statement, that is, verify assures that the proper data is replaced.

Format:

1	2–4	5–6	7–12	13–14	15–16	17–70	71–72	73–80
X'02'	VER		*sadd*		*esdid*	*flds*		nu

Where:

X'02'
must appear in column 1.

VER
is the verify control statement acronym.

sadd
is the hexadecimal starting address of the area to be verified.

esdid
is the External Symbol Identification, a number assigned by the compiler or assembler to the CSECT in which the verification occurs.

flds
is up to eleven 4-digit hexadecimal fields, separated by commas, each verifying previously loaded halfwords. A comma cannot follow the last field.

nu
means "not used" by the loader. This field may be blank or contain program identification.

Usage Notes:

1. The verify control statement must be in hexadecimal code.
2. The data in columns 17–70 (excluding commas) on the verify statement verifies what already was loaded into virtual storage beginning at the address specified in columns 7–12.
3. Place verify control statements immediately before corresponding replace control statements.
4. If the verify function fails:
   - The loaded system does not execute.
   - All following REP control statements for that CSECT are ignored until the next successful VER control statement for that CSECT.

Include Control Section Control Statement

The include control section statement changes the length of a CSECT. Use it only when replace control statements cause a control section to increase in length.

Format:

1	2–4	5–72
X'02'	ICS	*name size*

Where:

X'02'
must appear in column 1.

ICS
is the include control section statement acronym.

name
is the name of the CSECT you wish to enlarge.

size
is the total size required for the CSECT. If size is smaller than the assembled size, the assembled size is used.

Usage Notes:

1. Both arguments in the include control section statement are required.
2. The include control section statement must appear before the first ESD statement of the CSECT.

Delete Control Statement

The delete control statement deletes up to eight CSECTs.

Format:

1	2–4	5–72
X'02'	DEL	*name1* [... *name8*]

Where:

X'02'
must appear in column 1.

DEL
is the delete control section statement acronym.

name1 **[...*name8*]**
is up to eight CSECT names. These names must not be previously defined within the load. Subsequent attempts to define these names as part of the system or subsystem will result in those CSECTs being deleted.

INSTFPP EXEC

Use the INSTFPP EXEC to install licensed programs.

Format

The format of the INSTFPP EXEC is:

INSTFPP	[*prodspec1* [*prodspec2* ...*prodspecn*]] [(*options*[)]]
	options:
	[Prompt] [Memo]
	[NOPrompt] [NOMemo]
	[Install] [Rewind]
	[NOInstall] [NORewind]
	[All] [*rdev*]

:prodspec1 ... prodspecn
are the product specification codes that let you specify the products you want processed. These codes consist of the product number and the feature identification code as listed in the FEATURE$ PRODUCTS file. If no feature identification code exists for a product, specify just the product number. If one exists, attach it to the end of the product number. Specify these codes without imbedded hyphens or other punctuation. INSTFPP scans the files on the stacked tape and processes the selected licensed programs.

options:

Prompt
displays the prompts that ask if you want to process the specified licensed programs. PROMPT is the default.

NOPrompt
eliminates the prompts that asks if you want to install the specified licensed programs.

Memo
prints a product *Memo to Users* from the tape for each selected product. MEMO is the default.

NOMemo
lets you process the selected products without printing product *Memo to Users.* You cannot specify NOMEMO if you have specified NOINSTALL.

Install
lets you install the selected products. Install is the default.

NOInstall
lets you process the selected products without installing them. You cannot specify NOINSTALL if you have specified NOMEMO.

Rewind
makes sure the tape is rewound before and after product installation and that it is properly mounted. REWIND is the default.

NORewind
lets INSTFPP processing continue without tape rewinds before and after product installation. You can specify NOREWIND only if you invoke INSTFPP from the command line.

Note: Make sure the tape is properly mounted and attached as 181. You can only install products located after the initial tape position.

All
lets you process all the products on the tape. ALL is the default if you do not enter product specification codes.

rdev
lets you specify the real tape address if the tape is mounted on a tape drive but not attached to MAINT as 181.

Before Running INSTFPP

Before you invoke INSTFPP to install licensed programs:

1. Log on to the MAINT user ID.
2. Make sure that MAINT has all privilege classes. Enter:

```
access 291 c ■
xedit user direct ■
locate/USER MAINT ■
```

You should see a line similar to this:

```
USER MAINT NOLOG 16M 32M ABCDEFG
```

The last field in this line defines the privilege classes assigned to MAINT. If your installation uses the IBM-supplied privilege classes, you should have classes ABCDEFG. If you do not, type in any classes MAINT needs.

3. While you are still looking at MAINT's directory entry, check to see if the MAINT 319 minidisk is defined. If it is not, you must define it.
   a. File your directory:

```
file ■
```

   b. Map your USER DIRECT and examine the map to find a space for the 319 minidisk. Look for a gap of appropriate size and record the starting cylinder of that gap. The size you need depends on the number of program products you want to install.

```
diskmap user direct ■
File USER DISKMAP A has been created.
```

c. XEDIT your directory and add an entry for the new minidisk:

```
xedit user direct ■
```

```
locate/USER MAINT ■
locate/MDISK ■
```

Locate the MAINT minidisk definitions.

```
MDISK 319 devtype startcyl cyl label MW ALL ■
```

devtype and *label* are the device type and label of the DASD volume where the new minidisk is located. *startcyl* is the starting cylinder of the gap where the new minidisk is defined. *cyl* is the size of the new minidisk in cylinders.

```
■
```

Press **ENTER** to return to the command line.

```
file ■
```

4. If you changed the directory, either to authorize new privilege classes for MAINT or to define the MAINT 319 minidisk, issue DIRECTXA to update the directory and bring it online:

```
directxa user ■
VM/XA SYSTEM PRODUCT USER DIRECTORY CREATION PROGRAM - RELEASE 2.0
HCPDIR750W RESTRICTED PASSWORD FILE NOT FOUND
EOJ DIRECTORY UPDATED AND ONLINE
Ready; T=n.nn/n.nn hh:mm:ss
```

5. Link to the MAINT 319 minidisk in read/write mode:

```
link * 319 319 w ■
```

6. Format the 319 minidisk if you have just defined it:

```
format 319 p (blksize 2048 ■
DMSFOR603R FORMAT will erase all the files on disk P(319).
             Do you wish to continue?
             Enter 1 (YES) or 0 (NO).
1 ■
DMSFOR605R Enter disk label:
mnt319 ■
DMSFOR733I Formatting disk P
DMSFOR732I ?? cylinders formatted on P(319)
Ready; T=n.nn/n.nn hh:mm:ss
```

7. Access the 319 minidisk as P:

```
access 319 p ■
```

8. Verify that you are linked to the following minidisks:

   MAINT 190 (System disk)
   MAINT 191 (MAINT's work disk)
   MAINT 193 (CMS BASE)
   MAINT 194 (CP BASE)

The INSTFPP EXEC expects these disks to be accessed with these numbers. If your corresponding disks have different numbers, redefine and reaccess them. For example, if your system disk is 490 instead of 190, issue:

```
define 490 190 ■
access 190 fm ■
```

fm is any unused filemode. You cannot access the redefined system disk as S.

9. Verify that the INSTFPP code is on one of these disks. Enter:

```
listfile instfpp exec * ■
INSTFPP EXEC S2
Ready; T=n.nn/n.nn hh:mm:ss
```

10. Mount the optional feature product tape. INSTFPP stops if the tape is not mounted correctly.
11. Print a hard copy of your directory:

```
print user direct ■
```

Many product installation EXECs link to the user minidisks in write mode. If the link attempt fails, you may be asked to enter the write password of the minidisk.

12. Verify that you have temporary disk space available on a DASD volume of the same type as your system residence volume. You need at least 30 contiguous cylinders of 3380 temporary disk space or an equivalent amount of 3350 or 3375 disk space. Enter:

```
#cp query alloc
DASD rdevno volid type TDISK TOTAL=nnnn INUSE=nnnn AVAIL=nnnn
                       PAGE  TOTAL=nnnn INUSE=nnnn AVAIL=nnnn
                       SPOOL TOTAL=nnnn INUSE=nnnn AVAIL=nnnn
                       DRCT  TOTAL=nnnn INUSE=nnnn AVAIL=nnnn [,ACTIVE]
```

The system issues this response for each DASD volume attached to the system. ACTIVE indicates the DASD volume containing the active directory.

If necessary, use the CPFMTXA command to allocate more temporary disk space. See *VM/XA SP CMS Command Reference.*

13. Enter:

```
instfpp (noinstall memo ■
```

to print the product *Memo to Users* for reference and to copy the *Memo to Users* to the MAINT 319 minidisk.

14. Verify that appropriate spooling control options are in effect to direct the virtual printer spool files INSTFPP produces with the PRINT command to the desired real printer. Enter:

```
#cp query tag printer ■
PRT vdevno TAG:
tagtext
```

If you are not satisfied with this setting, change it. You may have to invoke the CP SPOOL command or the CP TAG command (or both).

If your printer handles only uppercase characters, use the FOLD option of the CP LOADBUF command. If your printer does not accept the LOADBUF command, print memos by issuing the PRINT command with the UPCASE option. In addition, if your printer cannot print special characters contained in the product memos printed by INSTFPP, look online at the product *Memo to Users* on the MAINT 319 minidisk.

Refer to *VM/XA SP CP Command Reference* and *VM/XA SP CMS Command Reference* for more information about these commands.

15. Set your virtual storage size to 16MB unless the product *Memo to Users* specifies otherwise.

```
define storage 16M ■
STORAGE = 16M
STORAGE CLEARED - SYSTEM RESET
ipl cms ■
VM/XA CMS 5.6 mm/dd/yy hh:mm
■
Ready; T=n.nn/n.nn hh:mm:ss
```

16. Mount the product tape on a tape drive and ready the tape drive.

    Warning: Do not attach the tape drive to MAINT. The INSTFPP EXEC will attach it.

Running INSTFPP

You can run INSTFPP in panel mode or by specifying products directly.

- If you want to install several licensed programs, or one licensed program for which you do not know the product specification code, you will find it easier to run INSTFPP in panel mode. Follow the instructions under "Using the INSTFPP Panels."
- If you want to install only a few licensed programs and know the product specification codes, or if you want to install all the licensed programs, you can run INSTFPP either way. If you prefer to specify products directly, go to "Specifying Products Directly" on page 608.

Using the INSTFPP Panels

1. Invoke INSTFPP with no arguments from a 3270 device (20 line minimum).
2. When a panel appears on your screen, enter the real tape drive address, change defaults if necessary, and press **ENTER**.
3. If you choose not to install all products on the product tape, enter an X next to the products you want to install.

Figures 39 through 45 on pages 605 through 608 show the INSTFPP panels.

```
SOINS01           INSTFPP INSTALLATION OPTIONS
------------------------------------------------------------------------------

If appropriate, change any defaults and then press the ENTER key.

      Real tape address (will be attached as 181):          ___
      Process all products on the tape (Y/N)?                 Y
      Be prompted before each product is processed (Y/N)?     Y

      Alternatives:
            (1) Install the product(s) and print the memo(s)
            (2) Only print the product memo(s)
            (3) Only install the product(s)
                                       Enter (1, 2, or 3):    1




------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=           5=          6=
 PF7=          8=          9=              10=          11=         12=Cursor

====>
```

Figure 39. INSTFPP Panel 1

Note: If the real tape address appears as R/W, your tape drive is attached to MAINT. Quit and detach the tape drive, then reissue INSTFPP.

```
SOINS02           INSTFPP PRODUCT SELECTION PANEL            Line 1 of 68
------------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5668854          ACF/Network Control Program
_  5664289          ACF/System Support Program
_  5664280          ACF/Virtual Telecommunications Access Method
_  5668899          APL2
_  5668808          Application Prototype Environment
_  5767032          Application System
_  5798RWL          Composition Utility
_  5664329          Contextual File Search/370 for VM/CMS
_  5664296          Cooperative Viewing Facility Version 2
_  5668813          Cross System Product/Application Development
_  5668814          Cross System Product/Application Execution
_  5668918          Cross System Product/Application Query
------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=          8=Forward   9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 40. INSTFPP Panel 2

```
SOINS02           INSTFPP PRODUCT SELECTION PANEL           Line 13 of 68
------------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5748XE4          Directory Maintenance
_  5748XXB          Display Management System for CMS
_  5664370          DisplayWrite/370
_  5748XX9          Document Composition Facility
_  5735XXB          Emulation Program
_  5654260          Environmental Recording Editing and Printing Program
_  5668890          Font Library Service Facility
_  5798DFH          FORTRAN Utilities
_  5668801          Graphical Data Display Manager - IMD
_  5668812          Graphical Data Display Manager - PGF
_  5668812    NL    Graphical Data Display Manager - PGF NL Feature
_  5664200          Graphical Data Display Manager Base
------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=Backward  8=Forward   9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 41. INSTFPP Panel 3

```
SOINS02           INSTFPP PRODUCT SELECTION PANEL           Line 25 of 68
------------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5664200    NL    Graphical Data Display Manager Base NL Feature
_  5668905          Graphical Display And Query Facility
_  5799AXX          Graphics Attachment Support Programming
_  5799BKE          Host Displaywriter Document Interchange
_  5668996          IBM BASIC/VM
_  5664185          IBM High-Accuracy Arithmetic Subroutine Library
_  5798DTE          IBM 3812 Pageprinter VM Support
_  5668897          Info Center/1
_  5668012          Interactive Instructional Presentation System
_  5664282          Interactive System Productivity Facility
_  5664285          Interactive System Productivity Facility/PDF
_  5664204    01    NetView Volume 1
------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=Backward  8=Forward   9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 42. INSTFPP Panel 4

```
SOINS02            INSTFPP PRODUCT SELECTION PANEL            Line 37 of 68
--------------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5664204    02    NetView Volume 2
_  5664204    03    NetView Volume 3
_  5734PL3          OS PL/1 Optimizing Compiler and Libraries
_  5734LM5          OS PL/1 Transient Library
_  5664293          Overlay Generation Language
_  5664199          Page Printer Formatting Aid
_  5796PNQ          Pascal/VS
_  5664298          PC Bond
_  5664312          Print Services Access Facility
_  5664198    B     Print Services Facility/VM Base
_  5664198    S     Print Services Facility/VM 3800
_  5664198    V     Print Services Facility/VM 3820
--------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=Backward  8=Forward   9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 43. INSTFPP Panel 5

```
SOINS02            INSTFPP PRODUCT SELECTION PANEL            Line 49 of 68
--------------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5664309          Professional Office System
_  5664309    PA    Professional Office System Applications Support Feature
_  5668AAA          Query Management Facility
_  5748XP1          RSCS Networking, Version 1
_  5664188          RSCS Networking, Version 2
_  5748XXJ          Structured Query Language/Data System
_  5664191          Virtual Machine Monitor Analysis Program
_  5664319          Virtual Machine/Personal Computer Host Server
_  5746AM2          Virtual Storage Extended/Virtual Storage Access Method
_  5664318          VM/Interactive Productivity Facility
_  5664364          VM Batch Facility
_  5798DMY          VM File Storage Facility
--------------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=Backward  8=Forward   9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 44. INSTFPP Panel 6

```
SOINS02          INSTFPP PRODUCT SELECTION PANEL          Line 61 of 68
------------------------------------------------------------------------
         Type an X next to the product(s) you want to install
         When you have  finished,  press the PF5 key to begin
         the installation process.

_  5748RC1          VM/Pass-through Facility
_  5796PNA          VM Real Time Monitor
_  5664283          VM/Integrated System-Productivity Facility
_  5664291          VMBACKUP Management System
_  5664292          VMTAPE Management System
_  5668958          VS COBOL II
_  5668806          VS FORTRAN
_  5664281          3270 PC File Transfer




------------------------------------------------------------------------
 PF1=Help      2=          3=Quit          4=Return      5=Execute   6=
 PF7=Backward  8=          9=Sort(prodid) 10=Sort(desc) 11=         12=Cursor

====>
```

Figure 45. INSTFPP Panel 7

Specifying Products Directly

If you enter arguments, the INSTFPP panels do not appear on your screen. You can specify product specification codes by listing them as you find them in the FEATURE$ PRODUCTS file. Omit hyphens or other punctuation marks, and leave one blank between each code. You can specify up to 130 characters, including the command and options, on the CMS command line.

To install all the licensed programs available, specify the ALL option.

After Running INSTFPP

After you run INSTFPP, do the following steps:

1. Execute manual installation and verification procedures as indicated in the product *Memo to Users* if necessary.
2. Tailor product-dependent files as indicated in the product *Memo to Users* if necessary.
3. Resave CMS if licensed programs that you installed loaded files to the MAINT 19E minidisk (the Y-disk).
4. Consider placing file directory information for shared, read/only minidisks into a DCSS using the SAVEFD command. Refer to *VM/XA SP CMS Command Reference.*

Notes:

1. INSTFPP cannot properly restore minidisks accessed as read/only extensions with a subset defined. INSTFPP reaccesses minidisks as read/only extensions with no subset specification.
2. INSTFPP leaves the tape drive containing the optional feature product tape attached as virtual address 181.
3. A console file is created during the installation process to record terminal activity and is spooled to the MAINT reader when INSTFPP is complete.

PROD LEVEL File

INSTFPP updates a PROD LEVEL file on the MAINT 319 minidisk with the results of each licensed program installation.

Example

Figure 46 shows an example of a PROD LEVEL file.

```
------------------------------------------------------------------------
5748XXB  Display Management System (DMS/CMS)
VER 1    REL 2     MOD 0
Time and date of entry: 20:25:25   30 Sep 1986
*** Product installed and verified successfully
------------------------------------------------------------------------
5799AXX  3277 Graphics Attachment Support PRPQ (GASP)
VER 1    REL 3     MOD 0
Time and date of entry: 21:17:25   30 Sep 1986
*** Product installed; manual verification required
```

Figure 46. Sample PROD LEVEL File

Messages

Each licensed program entry in the PROD LEVEL file has an update message associated with it. The possible update messages and their explanations are:

***** Product installed and verified successfully**

Explanation: The licensed program was installed correctly, and the product was verified successfully.

***** Product files loaded; see the Memo to Users to complete installation**

Explanation: The licensed program files have been loaded successfully. Refer to the product *Memo to Users* printed by INSTFPP. This memo tells you how to complete the installation of the product and then verify that it installed correctly. In some cases, the product *Memo to Users* refers to other documentation.

***** Product installed; manual verification required**

Explanation: The licensed program was installed, but it was not verified automatically. Refer to the product *Memo to Users* printed by INSTFPP. This memo tells you how to make sure the licensed program installed correctly. In some cases, the product *Memo to Users* refers to other documentation.

***** Product installed; verification failed**

Explanation: The licensed program was installed, but the automatic verification failed. Try to install the licensed program again after correcting any problems; if it does not verify correctly, contact your support personnel.

***** Product Installation EXEC failed; RC = *rc***

Explanation: The product installation EXEC failed, and the return code passed back by this EXEC to INSTFPP is *rc*. Refer to the product *Memo to Users* or product installation EXEC prolog to see what this return code means. If you cannot fix the problem, contact your support personnel.

Rerunning INSTFPP

If any licensed programs do not install correctly, take the following steps:

1. Try to solve the problem by using the console log, product *Memo to Users*, the PROD LEVEL file, and other product-specific documentation.
2. Ready the tape.
3. Invoke INSTFPP.
4. Reinstall products that did not install correctly, plus any products that have these products as prerequisites.

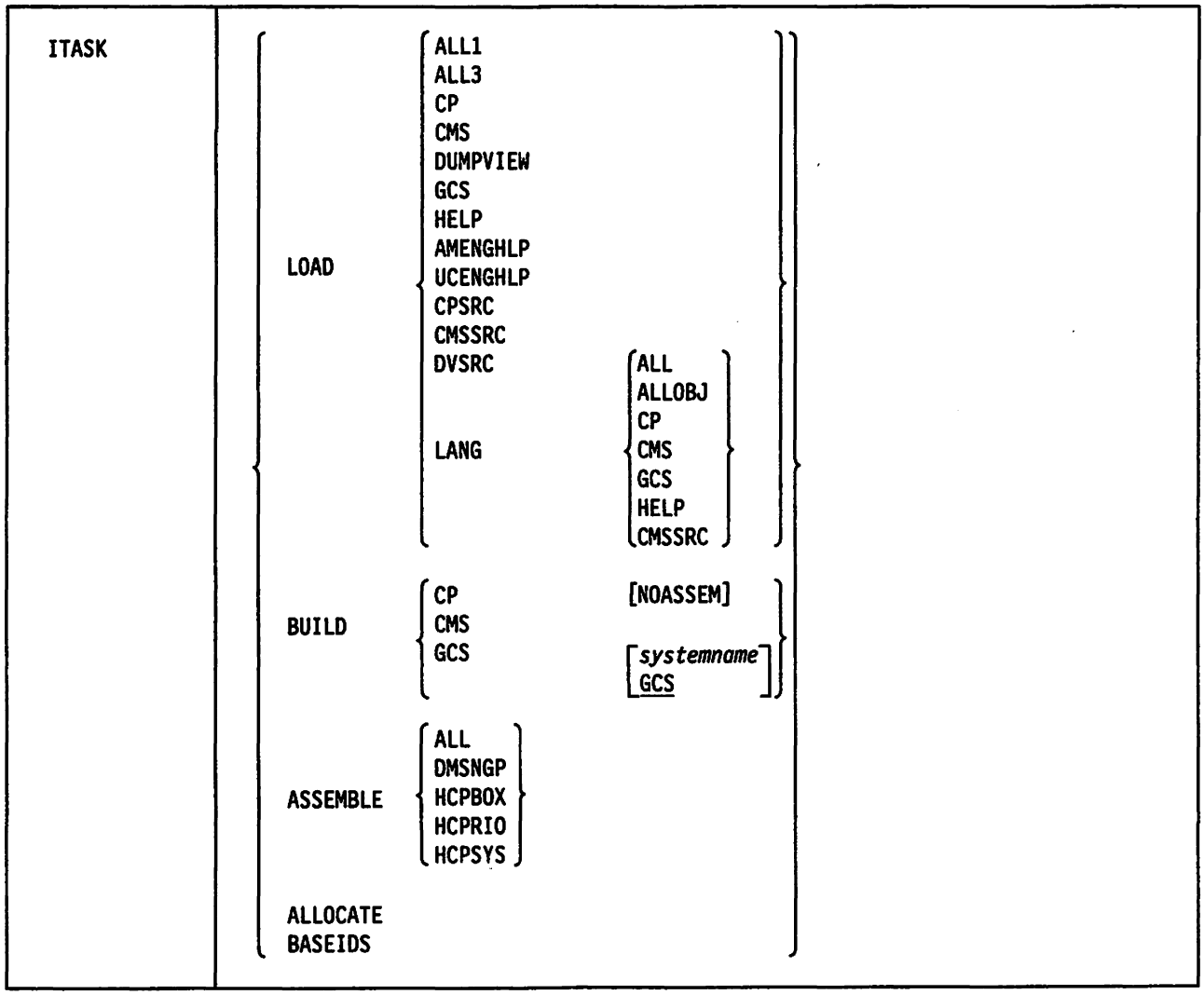
After you install each licensed program, follow the instructions in the product *Memo to Users* to verify that it has installed correctly (unless it was automatically verified during installation.)

ITASK EXEC

ITASK EXEC is a tool used primarily in the Starter System installation procedure. It invokes other EXECs and commands to perform installation and system generation tasks, to let you complete the installation process with fewer entries and decisions.

Format

The format of the ITASK EXEC is:



Where:

LOAD
invokes SPLOAD EXEC to load files from the product tapes, to the minidisks specified in SPLOAD PROFILE. Some LOAD operands may perform additional operations.

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

ALL1
loads all the CP and dump viewing facility OBJECT, DELTA, and APPLY files from Volume 1 of the product tape.

ALL3
loads all the CP, CMS, and dump viewing facility source files from Volume 3 of the product tape.

CP
checks to ensure that the SYSGEN tools are loaded, and then loads the the CP object code, DELTA, and APPLY files from the product tape.

CMS
checks to ensure that the SYSGEN tools are loaded, and then loads the IOCP files, CMS SYSTEM, CMS DELTA, CMS APPLY, and CMS BASE tape files from the product tape; then invokes the ASMGEND EXEC to build the starter system assembler.

DUMPVIEW
loads the dump viewing facility object, DELTA, and APPLY files from the product tape.

GCS
loads the GCS INTERFACE and GCS OBJECT tape files from the product tape.

HELP
loads the HELP FILES tape file from the product tape. First the mixed-case American English HELP files are loaded to the 19D disk; then they are copied to the 19C disk and converted to uppercase.

AMENGHLP
loads the mixed-case American English HELP files to the 19D disk.

UCENGHLP
loads the mixed-case American English HELP files directly to the 19C disk, then converts them to uppercase.

CPSRC
loads the CP SOURCE tape file from the product tape.

CMSSRC
loads the CMS SOURCE tape file from the product tape.

DVSRC
loads the DUMPVIEW SOURCE tape file from the product tape.

LANG
indicates that national language files are to be loaded from a national language feature tape.

ALL
loads the entire contents of the national language feature tape.

ALLOBJ
loads only the object (non-source) code. This consists of the following tape files: CP OBJECT, CMS BASE, and GCS OBJECT.

CP
loads only the CP OBJECT tape files.

CMS
loads only the CMS BASE tape files.

GCS
loads only the GCS OBJECT tape files.

HELP
loads only the HELP FILES tape file.

CMSSRC
loads only the CMS SOURCE tape file.

BUILD
invokes VMFBLD EXEC to build the specified nucleus.

CP
assembles the CP sample files; invokes the VMFBLD EXEC to build the CP nucleus.

If the NOASSEM option is used, the CP sample files are not assembled. This option assumes that these files are already assembled individually.

CMS
invokes the VMFBLD EXEC to build the CMS nucleus.

GCS
modifies a copy of the GCS loadlist (GCSLOAD EXEC) and changes the default configuration file entry (GCS) to match the filename of the GCS configuration file, *systemname* GROUP (if you do not specify a filename, the default is GCS); renames the filetype of the configuration file from GROUP to ASSEMBLE; assembles the configuration file; and invokes the VMFBLD EXEC procedure to build and save the GCS nucleus.

ASSEMBLE
assembles the specified sample file:

ALL
assembles the HCPBOX, HCPRIO, and HCPSYS files (but not the DMSNGP file).

DMSNGP
assembles only the DMSNGP file.

HCPBOX
assembles only the HCPBOX file.

HCPRIO
assembles only the HCPRIO file.

HCPSYS
assembles only the HCPSYS file.

ALLOCATE
loads in sample directory from the product tape, prompts for a new password to be used by both MAINT and Operator, replaces the starter system directory with this sample directory, and then loads the following files from Volume 1 of the product tape.

File	Contents
SYSGEN TOOLS	$DASD$ CONSTS, DISKMAP EXEC
SYSTEM SAMPLES	DVM PROFILE, SAMPNSS EXEC, starter IOCP, FILECONV SAMPEXEC, CONVSYS SAMPCMDS, CONVUCR SAMPCMDS, COMPSCAN SAMPEXEC, COMPSCAN SAMPLIST, VMFUECP EXEC, VMFUECMS EXEC, VMFUEDV EXEC

File	Contents
CPLOC SAMPLES	HCPBOX ASSEMBLE, HCPRIOXA ASSEMBLE, 33*nn* DIRECT, 33*nn* DISKMAP, HCPSYS*nn* ASSEMBLE
CMSLOC SAMPLES	DMSNGP ASSEMBLE, DMSNGP TEXT, DMSNGP TXTUCENG

After loading the files, the ITASK EXEC renames the following files:

Preload Name	Name After Load
33*nn* DIRECT	USER DIRECT
33*nn* DISKMAP	USER DISKMAP
HCPSYS*xx* ASSEMBLE	HCPSYS ASSEMBLE

The placement of these files is determined by the SPLOAD PROFILE.

The ITASK EXEC also allocates space on the DASD volumes identified in the $ALLOC$ user ID in the CP directory.

You are prompted for the real address of each DASD volume, and you have the option to SKIP any volume that you do not want to allocate. Allocation is done according to the entries in the directory; all space not specifically allocated is allocated as PERM.

BASEIDS
issues the CMS FORMAT command to format the remaining minidisks defined in the base CP directory which did not have code loaded to them during the installation process.

The minidisks formatted during this procedure are:

- AUTOLOG 191
- CMSBATCH 195
- DISKACNT 191
- EREP 191
- MAINT or XAMAINT 192, 194, 19C, 19D, 19E, 201, 291, 293, 294, 36E (PVM), 391, 392, 393, 394, 423, 490, 491, 492, 495, 49C, 49D, 501, 591, 592, 593, 594, 595, 596, 59E, 691, 692, 791, 892, 895, 896, 89E

  Note: If you issue ITASK BASEIDS from a user ID other than MAINT or XAMAINT, that user ID's minidisks are formatted instead.
- OPERATNS 191
- RSCS 191.

Also, a PROFILE EXEC is placed on the EREP 191 and DISKACNT 191 minidisks during this procedure.

Messages and Return Codes

DMSWTK002E **File** *fn ft* [*fm*] **not found [RC=28]**
DMSWTK003E **Invalid option:** *option* **[RC=24]**
DMSWTK008E **Device** *vdev* **invalid or nonexistent [RC=36]**
DMSWTK050E **Parameter missing after** *function* **[RC=24]**
DMSWTK070E **Invalid parameter** *parameter* **[RC=24]**
DMSWTK095E **Invalid address** *vstor* **[RC=100]**
DMSWTK360E **Invalid response** *response* **[RC=0]**
DMSWTK961E **There are class** *class* **files in the** *device*

DMSWTK965I **You may wish to tailor the following files before nuclei generation:**

HCPBOX ASSEMBLE
HCPRIO ASSEMBLE
HCPSYS ASSEMBLE
DMSNGP ASSEMBLE
USER DIRECT
[RC=20]

DMSWTK966R **Do you wish to have the HELP files converted to uppercase? [RC=0]**
DMSWTK967R **Type: (Yes) or No [RC=0]**
DMSWTK968I **The following minidisks defined in the base CP directory will be formatted:**

userid	**192**	*userid*	**49C**
userid	**194**	*userid*	**49D**
userid	**19C**	*userid*	**501**
userid	**19D**	*userid*	**591**
userid	**19E**	*userid*	**592**
userid	**201**	*userid*	**593**
userid	**291**	*userid*	**594**
userid	**293**	*userid*	**595**
userid	**294**	*userid*	**596**
userid	**391**	*userid*	**59E**
userid	**392**	*userid*	**691**
userid	**393**	*userid*	**692**
userid	**394**	*userid*	**791**
userid	**423**	*userid*	**892**
userid	**490**	*userid*	**895**
userid	**491**	*userid*	**896**
userid	**492**	*userid*	**89E**
userid	**495**		

[RC=0]

DMSWTK969I **A PROFILE EXEC has successfully been copied to {EREP's|DISKACNT's} 191 minidisk. [RC=0]**
DMSWTK970I **Formatting *user id vdevno* minidisk [RC=0]**
DMSWTK981R **What is the real address of your *volume* volume? [RC=0]**
DMSWTK982R **Type: Real address or SKIP [RC=0]**
DMSWTK983E **Violation of CMS naming convention found in *args* [RC=24]**
DMSWTK1310I **Do you want to format minidisks: AUTOLOG1 191, CMSBATCH 195, DISKACNT 191, *userid* 36E (PVM), EPEP 191, OPERATNS 191, and RSCS 191? [RC=0]**
DMSWTK8005R **Type: 1 (Yes) or 0 (No).**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference.*

The Patch Facility

The patch facility is used to make changes to TEXT files in the CP or CMS nucleus.

The patch facility lets you maintain (patch) object code when neither source nor object deck replacements are available. These patches are a temporary solution to object code problems until you receive a replacement file from IBM.

The procedure for maintaining object code is similar to that for maintaining source code. However, with object code, rather than changing source and reassembling the source file, you will apply changes directly to the TEXT files as the nucleus is generated.

Controlling Patches

You control patches with three kinds of files:

- Control files
- AUXiliary files
- Update files.

Note: As with source updates to base ASSEMBLE files, patches to TEXT files do not cause any changes to the original TEXT files because temporary TEXT files are created. The temporary TEXT files are used to generate the nucleus, then erased.

The following is a description of the files and elements you will need to understand to make object code patches:

Control File This file contains a MACLIB statement and a list of filetypes, one for each level of AUX file. It also contains the text deck qualifier for each level. A keyword, TX$, must follow the last AUX filename on a line to flag the AUX file or update as containing text patches.

Note: The patch facility creates a temporary control file with a filename of $$$TUP$$ CNTRL. The use of this name for any control file is restricted to the patch facility.

AUX File This file contains a list of the filetypes of update files. The update files contain source updates to source or updates to TEXT files.

Note: Any entry in the AUX file which contains the name of a patch update file must contain the TX$ control word between columns 8 and 13.

Load Map This function contains information from both the map of CSECT external symbol resolution and service level information. The map also includes local patches and co-requisite and prerequisite information. The date and time listed in the load map for each patched text file are the original date and time, not the date and time when the temporary deck was created.

TEXT File This file contains the data elements that have been assembled by the customer for source maintenance or provided (already assembled) by IBM for object maintenance. You use it to create executable modules. These files contain APAR corequisite and prerequisite information.

Note: The patch facility creates temporary text files with a filetype of TXTTUP. Only the HCPLDR EXEC (the loader EXEC) may use this filetype.

Example of a Patch Update File

The following are the functionally equivalent HCPLDR EXEC control statements contained in the update file, which will appear as comments to the update program:

```
./ * * CO-REQ: nnnnnnnn
./ * * PREREQ: nnnnnnnn
./ * NAME CSECTname
./ * *
./ * VER disp data
./ * REP disp data
./ * ICS name size
```

These statements are equivalent to the loader control statements in content. HCPLDR EXEC reformats the statements to make them acceptable to HCPLDR module.

Note: Pre-allocated patch areas are not required because the loader can expand any CSECT in the executable module as it is created from the text files. Addressability to the expanded area is available in every CSECT although space for a patch area need not be. This results in reduced size of executable modules because patch areas do not consume space until they are used.

The following is an explanation of each statement:

Note: Each statement below must be preceded by "./" .

*** NAME** *CSECTname*

This statement identifies which CSECT in a text file is to be patched. It is optional. The *CSECTname* must match the SD name from the ESD for the module. If you omit this statement, the CSECT with the same name as the filename is patched.

*** *** *comment*

This is a comment to the loader EXEC and is optional.

*** * CO-REQ:** *nnnnnnnn*

This is an APAR requisite comment. Use it to indicate dependencies on APARs or other patches. The following format is suggested:

```
./ * * CO-REQ: VM23418
./ * * PREREQ: VM23418
```

When there are no dependencies, enter NONE after the keywords CO-REQ: and PREREQ:.

*** VER** *disp data*

This statement verifies that the patch is applied at the correct point in the executable module. You need at least one verify statement for each patch. As much data as is required to assure uniqueness should be verified. For example, you may include a verify of the date in the copyright constant of the module prologue.

The *disp* is the displacement, in hex, of the location to be patched in the CSECT.

The *data* is the existing old data in hex to be replaced. This may be up to eleven 4-digit fields separated by commas. A comma may not follow the last halfword.

*** REP** *disp data*

This statement contains the data to replaced in the CSECT.

The *disp* contains the displacement in hex of the location to be patched in the CSECT.

The *data* is the new data in hex that will replace the existing data. This may be up to eleven 4-digit fields separated by commas. A comma may not follow the last halfword.

* ICS *name size*

This statement expands the CSECT in the executable module (nucleus). This statement is optional.

The *name* is the same name used in the NAME statement. If a NAME statement is not present, *name* must match the filename of the patch file in which it is included.

The *size* is the total size required for the CSECT.

The following are the functions used in the object service process:

- VMFBLD EXEC

  The VMFBLD EXEC invokes the loader EXEC (that is, HCPLDR EXEC).

- HCPLDR EXEC

  This EXEC exploits the Verify and Replace capabilities of the loader module.

  It gathers TEXT files into an executable nucleus based upon control information.

  Note: This function does not change previous HCPLDR command syntax except to include a NOPATCH option.

  This EXEC provides the same level of control for local service, when source is not available, as is currently provided for source file updates. This means that the nucleus-generation process automatically locates and tries to reapply a local patch each time a replacement update file arrives from IBM.

  This results in one of two possibilities:

  1. The local fix is applied and you receive replacement update files. You must then evaluate whether:
     - The local fix is obsolete.

       If the replacement update file contains the APAR fix that replaces the local fix, then you can remove the local fix by removing the object update from the AUX file or the control file.
     - The local fix is still needed.

       If the replacement update file does not contain an APAR fix that replaces the local fix, then you will leave the local fix in place.
  2. The local fix is not applied and you receive replacement service files. You must then evaluate whether:
     - The local fix is obsolete.

       If the replacement update file contains an APAR fix that replaces the local fix, you just accept the replacement deck.
     - The replacement update file does not contain an APAR fix equivalent to the local fix.

       If the replacement update file does not contain an APAR fix that replaces the local fix, you may:
       - Choose the APAR fix as more important, and just accept the replacement update file.
       - Fix both problems by accepting the replacement update file and reworking the local fix.

Compatibility with HCPLDR

A temporary composite TEXT file is created by the loader EXEC; the composite TEXT file is a copy of the IBM-supplied TEXT file with the patch update file merged into it. The temporary text deck will have the same date and time as the original. The NAME statement is not included in a composite TEXT file. The NAME statement is used only to identify the correct CSECT. When this statement is not present, the patch update file changes the CSECT with the same name as the TEXT file.

A composite TEXT file may include:

- The patch AUX file entry (included as a comment), placed following existing comments, but before the first ESD statement.
- An APAR requisite comment, placed after the AUX file comment, but before the first ESD statement.
- An optional ICS statement, placed after an APAR requisite statement, but before the first ESD statement.
- Required VER and REP statements, placed after any ICS statement, but before the first RLD statement or before the END statement if there is no RLD statement in the TEXT deck.
- Patch update file comments, included as they are encountered.

For each NAME statement in a patch update file and for each patch update file without a NAME statement, a VERIFY statement must be present or the patch is not accepted by the loader EXEC. The composite TEXT file will have the same filename and its filetype will be TXTTUP.

The loader EXEC works the same way as the current HCPLDR command works. The EXEC goes through the load list one module at a time, using the CNTRL file to determine the filetype. For each temporary composite TEXT file, the EXEC provides a filename with the filetype TXTTUP. In addition, the EXEC creates a $$$TUP$$ CNTRL file that contains the filetype qualifier (TUP that produces a filetype of TXTTUP) of the composite TEXT file created by the loader EXEC.

The loader EXEC invokes the HCPLDR MODULE by passing it a temporary control file named $$$TUP$$ along with other parameters. All temporary files such as TXTTUP or $$$TUP$$ have a filemode of 3; these files are erased automatically as soon as they are used.

Usage Notes

The patch facility is provided to give you:

- The same control that you have with source updates and IBM update file updates.
- The same tracking capability so that no previously applied patch will be lost or ignored when IBM replacement service is applied based upon the contents of the load map.

The following guidelines are recommendations to follow:

1. Keep each fix to a TEXT file in a separate update file.

   This also applies to source update fixes; each source update fix should be in a separate update file.

   Each fix should have an alphanumeric number that is the filetype of the update file.

2. Keep all local fix descriptions for the same TEXT file in the same AUX file, unless a fix applies to a different control file level.

   Local fixes for the same TEXT file should not be distributed over AUX files (different control file levels) arbitrarily. Local service should be easily distinguished from IBM service and should always be applied last. Local service can be distributed over separate control files for the purpose of maintaining different service levels with a single structure of AUX and update files. Each level can be built from a different control file containing only the desired level identifiers.

3. Never place local patches in AUX files from IBM. In other words, keep your local service separate from IBM service. Local service should be easily distinguished from IBM service and should always be applied last.

4. Patches to TEXT files should be applied only when no source file is available. Mixing source updates and TEXT file patches for the same module will lead to confusion and is not recommended.

Example of Local Service to TEXT Files

The following is an example of a control structure containing patches at more than one level. This is an appropriate use of the patch facility.

Filename	Filetype	Contents
PRODSYS	CNTRL	L3 AUXLCL3 TX$ P1 AUXP1 TEXT AUXXA
CMSSYS	CNTRL	L2 AUXLCL2 TX$ L3 AUXLCL3 TX$ P2 AUXP2 P1 AUXP1 TEXT AUXXA
TESTSYS	CNTRL	L1 AUXLCL1 TX$ L2 AUXLCL2 TX$ L3 AUXLCL3 TX$ P3 AUXP3 P2 AUXP2 P1 AUXP1 TEXT AUXXA
HCPXYZ	AUXLCL1	PATCH3 TX$ APAR3
HCPXYZ	PATCH3	./ * VER 24 4780,C204 ./ * REP 24 4700
HCPXYZ	AUXLCL2	PATCH2 TX$ APAR2
HCPXYZ	PATCH2	./ * VER 254 47F0,6062 ./ * REP 254 4700 ./ * VER 260 5810,7042,5010 ./ * REP 260 58F0,7042,50F0
HCPXYZ	AUXLCL3	PATCH1 TX$ APAR1
HCPXYZ	PATCH1	./ * VER 254 4740,6062 ./ * REP 254 47F0
HCPXYZ	TEXT	ESD TXT RLD END

Example of Local Service to ASSEMBLE Files

The following is an example of a control structure containing a temporary patch over a local source update. Although this works, it is not recommended. Whenever source code is available, you should use source updates instead of patches.

Filename	Filetype	Contents
XASYS	CNTRL	L1 AUXLCL1 TX$ P1 AUXP1 TEXT AUXXA
HCPXYZ	AUXLCL1	TEMP02 TX$ PROB1 LCFIX5 TX$ APAR2
HCPXYZ	TEMP02	./ * VER 24 4780,C204 ./ * REP 24 4700
HCPXYZ	LCFIX5	./ ADD 12340000 XYZLOOP TM FLAG,X'01' CHECK COMPLETE BO DONE EXIT LOOP
HCPXYZ	TXTL1	ESD TXT RLD END

PRELOAD MODULE

The preloader is a utility program that runs under CMS. It collects multiple text files and reformats them into a single text file. The function of the preloader is similar to that of a linkage editor, but the output is in standard text file format and does not include multiple CSECTS.

A program can be developed using separate assembly modules that reference each other. The preloader can then be used to combine the assembled text files into a single loadable text file.

Format

The format of the PRELOAD command is:

PRELOAD	*loadlist* [*control*]

Where:

loadlist
specifies the filename of an EXEC on the caller's A-disk or read-only extension containing records that define input to the preloader. Each of these records contains the filename and optionally the filetype of an input text file. The format of each loadlist record defining an input file must be:

```
&1 &2 filename  filetype
```

control
optionally specifies the filename of a CNTRL file residing on one of the caller's accessed disks. The format and interpretation of the CNTRL file is the same as that for the VM/SP VMFLOAD utility. It normally contains filetypes in priority sequence to be used for selecting input files if filetypes are not included in the loadlist file. The IBM-supplied control files are described in Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361.

Note: PRELOAD ignores records that have a PTF update level identifier. It then searches for the next lower level identifier to determine the filetype of the input text file. PRELOAD also ignores any options in the loadlist.

Input

The preloader gets input filenames from the loadlist. The filetype for each input file is determined in one of three ways:

1. If the loadlist record for a given input file includes a filetype entry, that filetype is used to locate the record.
2. If the loadlist record does not contain a filetype, and a "control" parameter was specified on the PRELOAD command line, the filetype constructed is in the format TXT*xxxxx*. In this case, *xxxxx* is the highest control level identifier in the control file for which a file can be located on the caller's accessed disks.
3. If no filetype is specified on the loadlist entry, and a control file has not been specified on the PRELOAD command line, then the default filetype value is TEXT.

Note: Input files are located by scanning the caller's disks in their access order. All input files must be on accessed disks.

Output

The preloader output consists of two files written to the caller's A-disk: *fn* TEXT and *fn* MAP.

The filename for each of these files is the same as that specified for the input loadlist file. If either of these files already exists on the caller's A-disk, the new copy replaces the old one.

TEXT File

The output TEXT file is a merged and linked composite of the input files. The first CSECT or private code section in the input expands to contain all input files. Its length is the sum of the lengths of the input files, rounded up to doubleword multiples between sections. Input TXT records of non-zero length are relocated and written to the output file.

The output RLD is a translated and relocated collection of all input RLD records. No sorting is done by the preloader. In general, each output ESD, TXT, and RLD entry appears in the same order as the corresponding input entry. ADCON and VCON fields are relocated within their TXT records. ORG statements that cause relocatable constant fields to overlay or be overlaid may cause results that differ from results obtained with a loader that completes TXT data loading prior to relocating ADCONs and VCONS.

MAP File

The output MAP file is a printable record of preloader processing, similar to a load map. The first line of the map contains:

- Output text filename
- Residence volume label and volume device address
- Date and time of file creation.

The next section of the map is a listing of the control file (if any) used. The remainder of the map contains, in processing order, a section for each input file. Each of these sections consists of:

- Filename, filetype, filemode of input file
- Residence volume label and virtual device address
- Input file's creation date and time
- Any invalid input records.

Messages

DMSPRE001E **No filename specified**
DMSPRE002E **File *fn ft fm* not found**
DMSPRE104S **Error *nn* reading file *fn ft fm* from disk**
DMSPRE105S **Error *nn* writing file *fn ft fm* on disk**
DMSPRE109S **Virtual storage capacity exceeded**
DMSPRE183E **Invalid {CONTROL|AUX} file control card**
DMSPRE234E **Error in LOAD LIST file *fn ft fm*[; no input]**
DMSPRE235E **Error *n* in input text file *fn ft* [*fm*]**
DMSPRE236E **Unresolved external reference(s) encountered**
DMSPRE237E **Duplicate external symbol(s) encountered**
DMSPRE238E **Preloader processing error**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference.*

SAMGEN EXEC

The SAMGEN EXEC creates a segment that contains the simulated VSE modules necessary to support Sequential Access Method (SAM) data management (DTFSD), the ESERV utility program, and Virtual Storage Extended/Virtual Storage Access Method (VSE/VSAM).

To install this segment, called CMSBAM, use the SAMGEN EXEC procedure.

Before you invoke SAMGEN EXEC, you need to define a virtual machine large enough to contain CMSBAM. To provide room for the loader tables, the size of the virtual machine should be at least 512K greater than the location at which you intend to save CMSBAM. However, you must not define your storage larger than the address at which you loaded CMSDOS.

SAMGEN EXEC performs the following operations:

- Fetches the simulated VSE phases from the CMSBAM DOSLIB file, which is supplied as part of VM
- Loads the simulated phases at the designated address
- Assigns a storage protection key of X'F'
- Saves the segment.

SAMPNSS EXEC

The SAMPNSS EXEC defines named saved systems.

Format

The format of the SAMPNSS EXEC is:

SAMPNSS	{ CMS CMSXA GCS CMSINST HELP CMSDOS CMSBAM CMSVSAM CMSAMS }

Where:

CMS CMSXA
creates named saved systems for CMS and CMSXA at pages E00 – FFF.

GCS
creates a named saved system for GCS at pages 0 – 6 (exclusive) and 400 – 5FF (shared).

CMSINST
creates a saved segment for CMSINST at pages C00 – C4F.

HELP
creates a saved segment for HELP at pages C50 – C9F in the INSTHELP segment space.

CMSDOS
creates a saved segment for CMSDOS at pages B00 – B0F in the DOSBAM segment space.

CMSBAM
creates a saved segment for CMSBAM at pages B10 – B3F in the DOSBAM segment space.

CMSVSAM
creates a saved segment for CMSVSAM at pages BA0 – BFF (shared) and A30 – AFF (exclusive) in the DOSBAM segment space.

CMSAMS
creates a saved segment for CMSAMS at pages B40 – B9F (shared) and A00 – A2F (exclusive) in the DOSBAM segment space.

The SAMPNSS EXEC:

1. Defines the named saved systems or segments that you request
2. Queries CP about the status of the named saved systems or segments
3. Displays the status of the named saved systems or segments as shown below:

```
OWNERID  FILE  TYPE  CL  RECS  DATE    TIME   FILENAME   FILETYPE
CMSMAINT nnnn   NSS  A   nnnn  mm:dd   hh:mm   nss       NSS
```

Messages

HCPNSD440I **{Named Saved System (NSS)|Saved segment}** *systemname* **was successfully defined in fileid** *fileno.*

HCPNSS440I **{Named Saved System (NSS)|Saved segment}** *systemname* **was successfully saved in fileid** *fileno.*
Unrecognized Parameters passed — *parm1 parm2*

Explanation: SAMPNSS detected an error in the specified parameter.

For information on error messages and codes, see *VM/XA SP System Messages and Codes Reference.*

SPLOAD EXEC

The SPLOAD EXEC loads files from the product tapes according to the load instructions contained in SPLOAD PROFILE. SPLOAD PROFILE is described in this section.

Format

The format of the SPLOAD EXEC is:

SPLOAD	*group element* [*fn*	* [*ft*	*]]

Where:

group element
is a tape file identifier that SPLOAD EXEC uses to locate the entry in SPLOAD PROFILE that contains the load instructions for this tape file.

fn [ft]
form a file specification template. By default, both are set to asterisks (*), meaning all files within a tape file. However, they may be set to a specific filename and/or filetype to selectively load specific files from a tape file to a minidisk.

SPLOAD PROFILE

In the sample SPLOAD PROFILE, data entries are organized by format number (see definition below), and format number sets are separated by comment lines. A comment line begins with a slash asterisk (/*). Each data entry in SPLOAD PROFILE has the following syntax:

```
group   element   userid   address   format   volume   fileno
```

Where:

group element
is a tape file identifier that identifies the profile entry containing the load instructions for this tape file.

userid
is the owner of the minidisk to which the tape file contents are loaded. This information is used only if SPLOAD EXEC must LINK and ACCESS the minidisk in order to load the file.

address
is the minidisk address to which the tape file is loaded.

format
is the tape format. SPLOAD PROFILE contains a complete load table for each tape format, describing the locations of the files on the product tapes. SPLOAD EXEC selects the appropriate entries in the profile depending on the user's tape format.

volume
is the product tape volume on which the particular tape file resides.

fileno
is the relative tape file number (location) of the specified tape file on the particular product tape volume.

```
!-------------------------------------------------------------!
!              6250 BPI VM/XA Merged Product Tape             !
!-------------------------------------------------------------!
!
! Volume 1
!
Memo      Files      MAINT     191   XASP20V1   1        2
Install   Tools      MAINT     191   XASP20V1   1        3
Sysgen    Tools      MAINT     193   XASP20V1   1        4
System    Samples    MAINT     191   XASP20V1   1        5
CPLOC     Samples    MAINT     295   XASP20V1   1        6
CMSLOC    Samples    MAINT     395   XASP20V1   1        7
CP        Object     MAINT     194   XASP20V1   1        8
CP        Putapply   MAINT     192   XASP20V1   1        9
CP        Putdelta   MAINT     294   XASP20V1   1       10
CP        Corapply   MAINT     291   XASP20V1   1       11
CP        Cordelta   MAINT     291   XASP20V1   1       12
DUMPVIEW  Object     MAINT     193   XASP20V1   1       13
DUMPVIEW  Putapply   MAINT     392   XASP20V1   1       14
DUMPVIEW  Putdelta   MAINT     293   XASP20V1   1       15
DUMPVIEW  Corapply   MAINT     391   XASP20V1   1       16
DUMPVIEW  Cordelta   MAINT     391   XASP20V1   1       17
!
! Volume 2
!
CMS       Base       MAINT     193   XASP20V2   2        2
CMS       Putapply   MAINT     392   XASP20V2   2        3
CMS       Putdelta   MAINT     293   XASP20V2   2        4
CMS       Corapply   MAINT     391   XASP20V2   2        5
CMS       Cordelta   MAINT     391   XASP20V2   2        6
IOCP      Files      MAINT     193   XASP20V2   2        7
CMS       System     MAINT     190   XASP20V2   2        8
GCS       Object     MAINT     595   XASP20V2   2        9
GCS       Interface  MAINT     193   XASP20V2   2       10
GCS       Putapply   MAINT     592   XASP20V2   2       11
GCS       Putdelta   MAINT     596   XASP20V2   2       12
GCS       Corapply   MAINT     491   XASP20V2   2       13
GCS       Cordelta   MAINT     491   XASP20V2   2       14
AMENGHLP  Files      MAINT     19D   XASP20V2   2       15
UCENGHLP  Files      MAINT     19C   XASP20V2   2       15
!
```

Figure 47 (Part 1 of 2): Sample SPLOAD PROFILE

```
! Volume 3
!
CP        Source     MAINT     394   XASP20V3   3        2
CMS       Source     MAINT     393   XASP20V3   3        3
DUMPVIEW  Source     MAINT     393   XASP20V3   3        4
!
!
/*---------------------------------------------------------------------*/
/*          6250 BPI VM/XA NLS Feature Tape                            */
/*---------------------------------------------------------------------*/
/*                                                                     */
CMS       Base       MAINT     193   R2M0NLS    1        2
CP        Object     MAINT     194   R2M0NLS    1        3
HELP      Files      MAINT     ?     R2M0NLS    1        4
CMS       Source     MAINT     393   R2M0NLS    1        5
DUMMY     Object     MAINT     XXX   R2M0NLS    1        6
GCS       Object     MAINT     595   R2M0NLS    1        7
/*                                                                   */
/*                                                                   */
```

Figure 47 (Part 2 of 2). Sample SPLOAD PROFILE

Warning: The disk addresses in the SPLOAD PROFILE, user directory, 56643089 $PPF, and SETUP EXEC, and the default addresses used by the ITASK EXEC **must all match**. If you change an address in any one of these places, you must change it in all of them. We recommend that you do not change any addresses.

Usage Notes

1. SPLOAD must do a tape rewind each time it is invoked to make sure that the proper product tape is mounted and to be able to position the tape to the proper tape file. If the incorrect tape is mounted then SPLOAD will prompt for the proper product tape.

   The first tape file on each product tape contains a header file named $TAPE$ HEADER. The first line of this file must be in the following format:

   `TAPENO = n ; TAPEFORMAT = f`

   where *n* is the relative tape number and *f* is the tape format (for example, R1M05500).

2. SPLOAD expects that the tape drive being used is ATTACHed at virtual address 181. If not, SPLOAD will not be able to continue.

3. SPLOAD will ACCESS the target minidisk at an unused mode letter nearest the end of the alphabet. It will use Z if all modes are used.

4. SPLOAD restores the minidisk access order to its original state just before exiting. However, SPLOAD will not be able to recreate mode extension situations.

5. If SPLOAD does a LINK to a minidisk, followed by an ACCESS to that disk and receives a return code of 100 from the ACCESS, an attempt will be made to FORMAT the disk.

   SPLOAD will provide a minidisk label to FORMAT which will consist of the first 3 characters of the user ID which owns the minidisk, concatenated with a 3-digit minidisk address. If the minidisk address is four digits long, then only the first 2 characters of the user ID will be used.

   Note: For user ID MAINT, the default characters used will be either MNT or MT, depending on the size of the minidisk address.

Messages and Return Codes

DMSWTK002E *fn ft* **not found [RC=28]**
DMSWSL032E **Invalid filetype** *ft* **[RC=24]**
DMSWSL050E **Parameter missing after** *value* **[RC=24]**
DMSWSL062E **Invalid character** *char* **in fileid** *fn* **[RC=20]**
DMSWTK070E **Invalid parameter** *parameter* **[RC=24]**
DMSWSG095E **Invalid address** *vaddr* **[RC=100]**
DMSWSL252E **Invalid filename** *filename* **[RC=24]**
DMSWSL409I **Loading** *fn ft* **to the** *vdev* **disk attached to** *user id* **[RC=0]**
DMSWSL737R **Enter the minidisk address for the** *group element* **[RC=0]**
DMSWSL963E *keyword value* **not found in** *fn ft fm* **[RC=24]**
DMSWSL964R **Wrong tape mounted; mount product tape** *n*
Press ENTER when the correct tape is mounted or type QUIT [RC=0]
DMSWSL986I **Unable to restore ACCESS to** *mdisk* **[RC=0]**

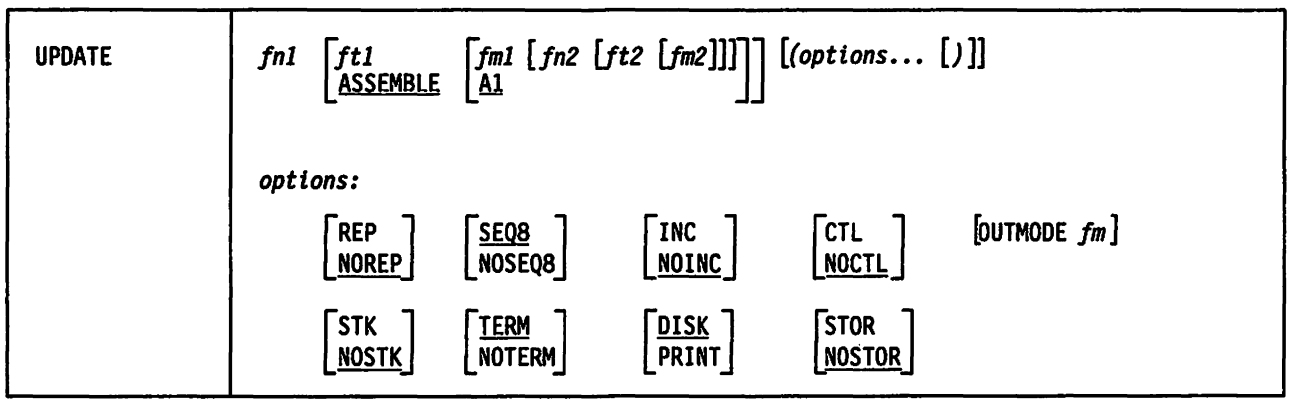
For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference.*

UPDATE Command

Use the UPDATE command to modify program source files. The UPDATE command accepts a source input file and one or more files containing UPDATE control statements and updated source records; then it creates an updated source output file, an update log file indicating what changes, if any, were made, and an update record file if more than a single update file is applied to the input file.

Format

The format of the UPDATE command is:



Where:

fn1 ft1 fm1
is the file identifier of the source input file. The file must consist of 80-character card image records with sequence fields in positions 73 – 80 or 76 – 80. If the filetype or filemode is omitted, ASSEMBLE and A1 are assumed, respectively.

fn2 ft2 fm2
is the file identifier of the update file. If the NOCTL option is in effect, this file must contain UPDATE control statements and updated source records. The default file identifier is *fn1* UPDATE A1. If the CTL option is specified, this file must be a control file that lists the update files to be applied; the default file identifier is *fn1* CNTRL A1.

***options*:**

REP
creates an output source file with the same filename as the input source file. If the output file is placed on the same disk as the input file, the input file is erased.

NOREP
retains the old file in its original form, and assigns a different filename to the new file, consisting of a dollar sign ($) plus the first 7 characters of the input filename (*fn1*).

SEQ8
specifies that the entire sequence field (columns 73 – 80) contains an 8-digit sequence number on every record of source input.

NOSEQ8
specifies that columns 73 – 75 contain a 3-character label field, and that the sequence number is a 5-digit value in columns 76 – 80.

Note: Source files sequenced by the CMS editor are sequenced, by default, with 5-digit sequence numbers.

INC
increases sequence numbers in columns 73 – 80 in each record inserted into the updated output file, according to specifications in UPDATE control statements.

NOINC
puts asterisks (********) in the sequence number field of each updated record inserted from the update file.

CTL
specifies that *fn2*, *ft2*, and *fm2* describe an update control file for applying multiple update files to the source input file.

Note: The CTL option implies the INC option.

NOCTL
specifies that a single update file is to be applied to the source input file.

OUTMODE *fm*
specifies that UPDATE writes the files it creates on the fm disk. UPDATE writes the files as outlined in the note below. If you do not specify a filemode when using OUTMODE, the filemode number defaults to 1. This disk must be an accessed CMS read/write disk or UPDATE terminates.

Note: UPDATE takes the following steps to determine the disk upon which it places output files. The search stops as soon as one of the following steps is successful:

1. If you specify the OUTMODE option, then UPDATE places the output files on the specified disk. That disk must be read/write. If you accessed the disk as a read only extension, then UPDATE displays the following message:

   ```
   DMSUPD037E DISK fm IS READ-ONLY
   ```

2. If the disk on which the original source file resides is read/write, then UPDATE places the output files on that disk.
3. If that disk is a read-only extension of a read/write disk, then UPDATE places the output files on that particular read/write disk.
4. If neither of the other steps is successful, then UPDATE places the output files on the primary read/write disk (the A-disk).

STK
stacks information from the control file in the CMS console stack. STK is valid only if the CTL option is also specified and is useful only when the UPDATE command is executed in an EXEC procedure.

NOSTK
does not stack control file information in the console stack.

TERM
displays warning messages at the terminal whenever a sequence or update control card error is discovered. (Such warning messages appear in the update log whether or not they are displayed at the terminal.)

NOTERM
suppresses the display of warning messages at the terminal. However, error messages that terminate the entire update procedure are displayed at the terminal.

DISK
places the update log file on disk. This file has a file identifier *fn* UPDLOG, where *fn* is the filename of the file being updated.

PRINT
prints the update log file directly on the virtual printer.

STOR
specifies that the source input file is to be read into storage and the updates performed in storage prior to placing the updated source file on disk. This option is meaningful only when used with the CTL option since the benefit of increased processing speed is realized when processing multiple updates. STOR is the default when CTL is specified.

NOSTOR
specifies that no updating is to take place in storage. NOSTOR is the default when single updates are being applied (CTL is omitted from the command line).

Update Control Statements

The UPDATE control statements let you insert, delete, and replace source records, as well as resequence the output file.

All references to the sequence field of an input record refer to the numeric data in columns 73−80 of the source record, or columns 76−80 if NOSEQ8 is specified. Leading zeros in sequence fields are not required. If no sequence numbers exist in an input file, a preliminary UPDATE with only the "./ S" control statement can be used to establish file sequencing.

Sequence numbers are checked while updates are being applied; an error condition results if any sequence errors occur in the update control statements, and warnings are issued if an error is detected in the sequencing of the input file. Any source input records with a sequence field of eight blanks are skipped, without any indication of a sequence error. Such records may be replaced or deleted only if they occur within a range of records that are being replaced or deleted entirely and if that range has limits with valid sequence numbers. There is no means provided for specifying a sequence field of blanks on an UPDATE control statement.

Control Statement Formats:

All UPDATE control statements are identified by the characters "./" in columns 1 and 2 of the 80-byte record, followed by one or more blanks and additional, blank-delimited fields. Control statement data must not extend beyond column 50.

SEQUENCE Control Statement

Use the sequence control statement to resequence the updated source output file in columns 73−80 (if SEQ8 is specified), or in columns 76−80 with the label placed in columns 73−75 (if NOSEQ8 is specified).

The format of the SEQUENCE control statement is:

```
./ S [seqstrt [seqincr [label]]]
```

Where:

seqstrt
is a 1- to 8-digit numeric field specifying the first decimal sequence number to be used. The default value is 1000 if SEQ8 is specified and 10 if NOSEQ8 is specified.

seqincr
is a 1- to 8-digit numeric field specifying the decimal increment for resequencing the output file. The default is the *seqstrt* value.

label
is a 3-character field to be duplicated in columns 73 – 75 of each source record if NOSEQ8 is specified. The default value is the first 3 characters of the input filename (*fn1*).

If you use the SEQUENCE statement, it must be the first statement in the update file. If any valid control statement precedes it, the resequence operation is suppressed.

When the sequence control statement is the first statement processed, the sequence numbers in the source file are checked and a warning message is issued for any errors. If the sequence control statement is processed after updates have been applied, no warning messages will be issued.

Each source record, including unchanged records from the source file and records inserted from the update file, is resequenced in columns 73 – 80 as it is written onto the output file.

INSERT Control Statement

Use the insert control statement to insert all records following it, up to the next control statement, into the output file.

The format of the INSERT control statement is:

```
./ I seqno [$ [seqstrt [seqincr]]]
```

Where:

seqno
is the sequence number of the record following which new records are to be added.

$
is an optional delimiter indicating that the inserted records are to be sequenced by increments.

seqstrt
is a 1- to 8-digit numeric field specifying the first decimal number to be used for sequencing the inserted records.

seqincr
is a 1- to 8-digit numeric field specifying the decimal increment for sequencing the inserted records.

All records following the "./ I" statement, up to the next control statement, are inserted in the output file following the record identified by the *seqno* field. If the NOINC option is specified, each inserted record is identified with asterisks (********) in columns 73 – 80. If either the INC or CTL option is specified, the records are inserted unchanged in the output file, or they are sequenced according to the *seqstrt* and *seqincr* fields, if the dollar sign ($) key is specified.

The default sequence increment, if the dollar sign is included, is determined by using one tenth of the least significant, nonzero digit in the *seqno* field, with a maximum of 100. The default *seqstrt* is computed as *seqno* plus the default *seqincr*. For example, the control statement:

```
./ I 2600 $ 2610
```

causes the inserted records to be sequenced XXX02610, XXX02620, and so forth (NOSEQ8 assumed here). For the control statement:

```
./ I 240000 $
```

the defaulted *seqincr* is the maximum, 100, and the starting sequence number is 240100. SEQ8 is assumed, so the inserted records are sequenced 00240100, 00240200, and so forth.

If either INC or CTL is specified but the dollar sign ($) is not included, whatever sequence number appears on the inserted records in the update file is included in the output file.

DELETE Control Statement

Use the delete control statement to delete one or more records from the source file.

The format of the DELETE control statement is:

```
./ D seqno1 [seqno2] [$]
```

Where:

seqno1
is the sequence number identifying the first or only record to be deleted.

seqno2
is the sequence number of the last record to be deleted.

$
is an optional delimiter indicating the end of the control fields.

All records of the input file, beginning at *seqno1*, up to and including the *seqno2* record, are deleted from the output file. If the *seqno2* field is omitted, only a single record is deleted.

REPLACE Control Statement

Use the replace control statement to replace one or more input records with updated records from the update file.

The format of the REPLACE control statement is:

```
./ R seqno1 [seqno2] [$ [seqstrt [seqincr]]]
```

Where:

seqno1
is the sequence number of the first input record to be replaced.

seqno2
is the sequence number of the last record to be replaced.

$
is an optional delimiter key indicating that the substituted records are to be sequenced incrementally.

seqstrt
is a 1- to 8-digit numeric field specifying the first decimal number to be used for sequencing the substituted records.

seqincr
is a 1- to 8-digit numeric field specifying the decimal increment for sequencing the substituted records.

All records of the input file, beginning with the *seqno1* record, up to and including the *seqno2* record, are replaced in the output file. They are replaced with the records that follow the "./ R" statement in the update file, and precede the next control statement.

As with the "./ D" (delete) function, if the *seqno2* field is omitted, only a single record is replaced, but it may be replaced by more than a single inserted record. The "./ R" (replace) function is performed as a delete followed by an insert: thus, the number of statements inserted need not match the number deleted. The dollar sign ($), *seqstrt*, and *seqincr* processing is identical to that for the insert function.

COMMENT Statement

Use the comment statement to insert comments into the file.

The format of the COMMENT statement is:

```
./ * [comment]
```

Where:

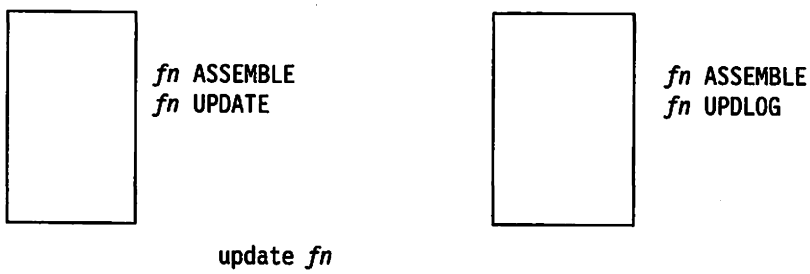
*
indicates that this is a comment statement and is only copied into the update log file.

Summary of Files Used by the UPDATE Command

The following discussion shows input and output files used by the UPDATE command for a:

- Single-level update
- Multilevel update
- Multilevel update with an auxiliary control file.

Single-Level Update



Where:

***fn* ASSEMBLE**
is the source input file.

***fn* UPDATE**
contains UPDATE control statements and updated source input records.

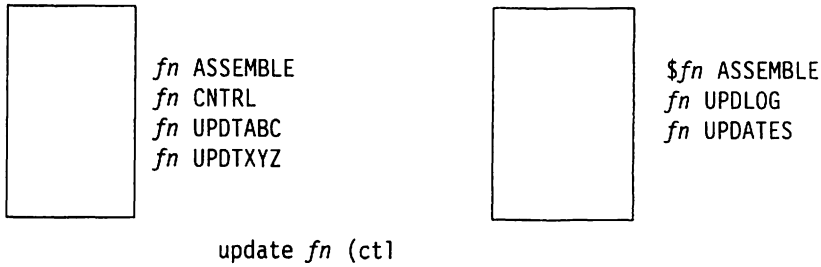
$*fn* ASSEMBLE
is the updated source file, incorporating changes, additions, and deletions specified in the update file. The output filetype is always the same as the filetype of the input file. These default filetypes and filemodes can be overridden on the command line; for example:

```
update testprog cobol b fix cobol b (rep ■
```

results in a source file TESTPROG COBOL B being updated with control statements contained in the file FIX COBOL B. The output file replaces the existing TESTPROG COBOL B.

***fn* UPDLOG**
contains a record of updates applied. If you do not want this file written on disk, specify the PRINT option.

Multilevel Update



Where:

***fn* ASSEMBLE**
is the source input file.

***fn* CNTRL**
is the control file that lists updates to be applied to the source file. These default filetypes and filemodes can be overridden on the command line; for example:

```
update acct pliopt a test cntrl a (ctl ■
```

results in the file TEST CNTRL being used by the UPDATE command to locate the update files for ACCT PLIOPT.

***fn* UPDTABC**
***fn* UPDTXYZ**
are update files containing UPDATE control statements and new source records. These files must have filenames that are the same as the source input file. The first 4 characters of the filetype must be UPDT. The UPDATE command searches all accessed disks to locate the update files.

$*fn* ASSEMBLE
is the updated source file, incorporating changes, additions, and deletions specified in the update files. The filetype is always the same as the filetype of the source input file.

***fn* UPDLOG**
contains a record of updates applied. If you do not want this file written on disk, specify the PRINT option.

***fn* UPDATES**
summarizes the updates applied to the source file.

The CONTROL FILE (*fn* CNTRL) may not contain UPDATE control statements. It may only list the filetypes of the files that contain UPDATE control statements. This control file contains the records:

```
TEXT MACS maclibname
TWO UPDTABC
ONE UPDTXYZ
```

where UPDTABC and UPDTXYZ are the filetypes of the update files. The UPDATE command applies these updates to the source file beginning with the last record in the control file. Thus, the updates in *fn* UPDTXYZ are applied before the updates in *fn* UPDTABC.

When you create update files whose filetypes begin with UPDT, you may omit these characters when you list the updates in the control file; thus, the CNTRL file may be written:

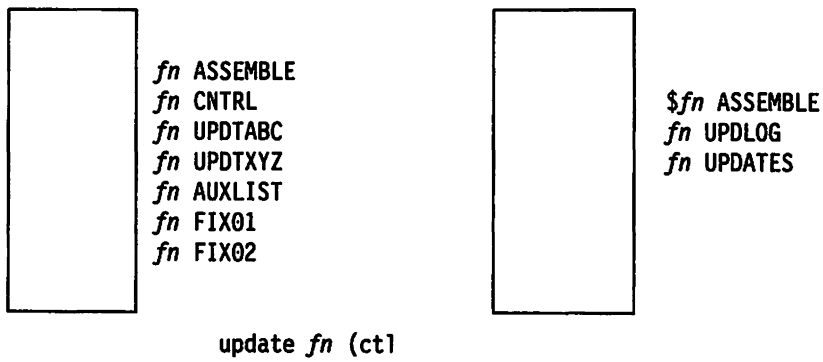
```
TEXT MACS maclibname
TWO ABC
ONE XYZ
```

TEXT, TWO, ONE: The first column of the control file consists of an update level identifier, which may be from 1 to 5 characters long. These identifiers are used by VM updating procedures, like the VMFHASM EXEC, to locate and identify text decks produced by multilevel updates.

MACS: The first record in the control file must be a MACS record that contains an update level identifier (TEXT) and, optionally, lists up to nine macro library (MACLIB) filenames.

UPDATE uses the information provided in the MACS card and the update level identifier only when the STK option is specified. This information is, however, required in the CNTRL file.

Multilevel Update with Auxiliary Control File



***fn* ASSEMBLE, *fn* CNTRL, *fn* UPDTABC, *fn* UPDTXYZ, $*fn* ASSEMBLE, *fn* UPDLOG,** and **fn UPDATES** are used as described for "Multilevel Update", except that the CNTRL file contains:

```
TEXT MACS maclibname
TWO UPDTABC
ONE UPDTXYZ
TEXT AUXLIST
```

AUX in the filetype AUXLIST indicates that this is the filetype of an auxiliary control file that contains an additional list of updates. The first 3 characters of the filetype of an auxiliary control file must be AUX; the remaining characters (to a maximum of 5 characters) may be anything. The filename must be the same as the source input file.

An auxiliary file may also be specified as:

```
xxxxx AUX
```

in the control file. For example, the record:

```
FIX TEST AUX
```

identifies the auxiliary file *fn* AUXTEST.

Note: If you give an auxiliary control file the filetype AUXPTF or an update level identifier of AUX, the UPDATE command assumes that it is a simple update file and does not treat it as an auxiliary file.

Preferred AUX File

A preferred AUX file may be specified. A preferred AUX file contains the version of an update that applies to your version of the source file. (There may be more than one version of the same update if there is more than one version of the source file. For example, you need one version for the source file that has a system extension program product installed, and you need another version for the source file that does not have a program product installed.)

When you specify an auxiliary control file, you can specify more than one filetype. The first filetype indicates a file that UPDATE uses only on one condition: the files that the second and subsequent filetypes indicate do not exist. If they do exist, this AUX file entry is ignored and no updating is done. The files that the second and subsequent filetypes indicate are preferred because, if they exist, UPDATE does not use the file that the first filetype indicates. For example, assume that the file *fn* ASSEMBLE does exist. The control file MYMODS CNTRL:

```
TEXT  MACS  MYMACS  maclibname
MY2   AUXTEST
MY1   AUXMINE AUXTEST
```

and the command:

```
update fn assemble * mymods cntrl (ctl ■
```

would result in UPDATE finding the preferred auxiliary control file *fn* AUXTEST, and therefore not using *fn* AUXMINE to update *fn* ASSEMBLE. UPDATE would then proceed to the MY2 AUXTEST entry and update *fn* ASSEMBLE with the updates listed in *fn* AUXTEST. It is assumed that AUXTEST and AUXMINE list similar, but mutually exclusive, updates.

The search for a "preferred" AUX file will continue until one is found or until the token is an invalid filetype; that is, less than 4 or more than 8 characters. This token and the remainder of the line are considered a comment.

fn FIX01 and *fn* FIX02 are update files containing UPDATE control statements and new source records to be incorporated into the input file. When update files are listed in an auxiliary control file, they can have any filetype you choose, but the filename must be the same as the source input file. The update files, as well as the AUX file, may be on any accessed disk. These are indicated in *fn* AUXLIST as follows:

```
FIX02
FIX01
```

The updates are applied from the bottom of the auxiliary control file. Thus, *fn* FIX01 is applied to the source file before *fn* FIX02. Since the auxiliary control file is listed at the bottom of the control file, these updates are applied before UPDTXYZ and UPDTABC.

Additional Control File Records

In addition to the MACS record, the filetypes of update (UPDT) files, and the filetypes of auxiliary control (AUX) files, a control file may also contain:

- Comments. These records begin with an asterisk (*) in column 1. Comments are also valid in AUX files.
- PTF records. If the characters PTF appear in the update level identifier field, the UPDATE command expects the second field to contain the filetype of an update file. The filetype may be anything; the filename must be the same as the source input file.
- Update level identifiers not associated with update files.

The following example of a control file shows all the valid types of records:

```
* Example of a control file
ABC MACS MYLIB
TEXT
004  UPDTABC
003  XYZ
002  AUXLIST1
001  LIST2  AUX
PTF  TESTFIX
```

The STK Option

The STK (stack) option is valid only with the CTL option and is meaningful only when the UPDATE command is invoked within an EXEC procedure.

When the STK option is specified, UPDATE stacks the following data lines in the console stack:

```
first line:  * update level identifier
second line: * library list from MACS record
```

The update level identifier is the identifier of the most recent update file that was found and applied. For example, if a control file contains:

```
TEXT MACS DMSGPI DMSOM OSMACRO OSPSI TESTMAC
OFA  UPDTOFA
PFA  UPDTOFA
```

and the UPDATE command appears in an EXEC as follows:

```
UPDATE SAMPLE (CTL STK &READ VARS &STAR &TEXT &READ VARS
&STAR &LIB1 &LIB2 &LIB3 &LIB4 &LIB5 &LIB6
```

then the variable symbols set by the &READ VARS statements have the following values if the file SAMPLE UPDTOFA is found and applied to the file SAMPLE ASSEMBLE:

Symbol	Value
&STAR	*
&TEXT	OFA
&LIB1	DMSGPI
&LIB2	DMSOM
&LIB3	OSMACRO
&LIB4	OSPSI
&LIB5	TESTMAC
&LIB6	null

The library list may be useful to establish macro libraries in a subsequent GLOBAL command within the EXEC procedure. If no update files are found, UPDATE stacks the update level identifier on the MACS record.

Message

FILE *fn ft fm*, REC #*n* = update control statement

Explanation: This message is displayed when the TERM option is specified and an error is detected in an update file. It identifies the file and record number where the error is found.

Return Codes

The UPDATE command issues the following return codes:

Return Code	Possible Causes
4	• No filename specified • Input file sequence error
8	• Sequencing overflow • Sequence increment is zero • Sequence error introduced in output file
12	• Premature EOF of file; sequence number not found • Missing PTF file • ./s not first card in input file; card ignored • Invalid char in sequence field • Sequence number not found • Invalid update file control card
24	• Invalid option • Invalid mode • Option specified twice • Conflicting options • Invalid parameter • Option STK invalid without CTL
28	• File not found • File UPDATE CMSUT1 *fm* already exists
32	• File is not fixed, 80-character records • Missing or duplicate MACS card in control file • Invalid file control card
36	• Disk is read-only • Disk not accessed
40	• No update files were found
41	• Insufficient storage to begin update • Insufficient storage to complete update
100	• Error reading file from disk • Error writing file on disk

UTILITY EXEC

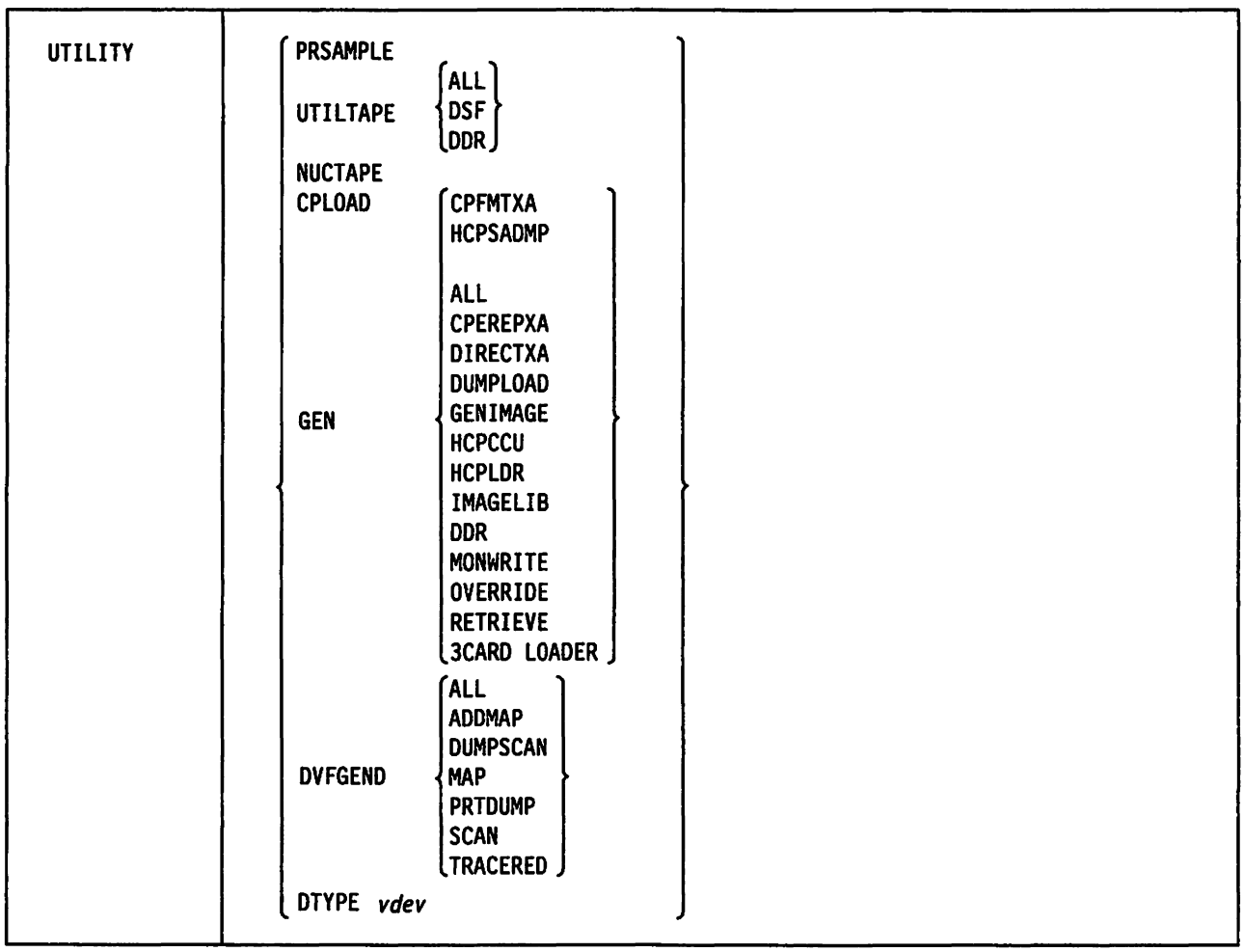
The UTILITY EXEC provides occasionally used installation utility functions:

- Printing the system definition files
- Creating a stand-alone service utility tape for either or both DSF and DDR
- Writing a backup IPLable copy of the CP nucleus to tape
- Creating CP load list
- Generating the following utilities:

```
ADDMAP      CPEREPXA    CPFMTXA     DDR
DIRECTXA    DUMPLOAD    DUMPSCAN    GENIMAGE
HCPCCU      HCPLDR      HCPSADMP    IMAGELIB
IPL DDRXA   MAP         MONWRITE    OVERRIDE
PRTDUMP     RETRIEVE    SCAN        TRACERED
3CARD LOADER
```

Format

The format of the UTILITY EXEC is:



Where:

PRSAMPLE
prints the following system definition files:

- USER DIRECT
- DMSNGP ASSEMBLE
- HCPRIO ASSEMBLE
- HCPSYS ASSEMBLE.

These files are spooled to the system printer. The minidisks containing these files must be accessed before using the PRSAMPLE operand.

UTILTAPE
creates a tape containing any or all of the following stand-alone utility programs:

- Device Support Facility (DSF)
- DASD Dump/Restore program (DDRXA).

Each IPLable program is written as a separate tape file, with one or more tape marks separating the programs. Therefore, when you IPL a program off the tape, you may have to IPL more than once to bypass the tape marks.

NUCTAPE
writes a backup IPLable copy of the CP nucleus to tape.

CPLOAD
creates the CP load list, which has a file name of CPLOAD and a file type of EXEC.

GEN
creates any of the following stand-alone utility programs on disk from their associated object modules (text decks):

- CPEREPXA MODULE, the error recording module
- CPFMTXA MODULE, the module that CP-formats minidisks for CP use
- DIRECTXA MODULE, the user directory program
- DUMPLOAD MODULE
- GENIMAGE MODULE
- HCPCCU MODULE
- CMS version of the system loader (HCPLDR MODULE) and the stand-alone version of the system loader (HCPLDR LOADER)
- Stand-alone dump program (HCPSADMP)
- IMAGELIB MODULE
- 370-XA version of the DASD DUMP RESTORE program (IPL DDRXA)
- DDR MODULE
- MONWRITE MODULE
- OVERRIDE MODULE
- RETRIEVE MODULE
- 3 CARD LOADER.

UTILITY GEN ALL creates all the programs listed **except** CPFMTXA MODULE and HCPSADMP.

DVFGEND
creates any or all of the following modules:

- ADDMAP MODULE
- DUMPSCAN MODULE
- MAP MODULE
- PRTDUMP MODULE
- SCAN MODULE
- TRACERED MODULE.

DTYPE *vdev*

issues a DIAGNOSE code X'24' for the specified virtual device address, uses the returned information to look up the DASD type in the $DASD$ CONSTS file, and returns that information to the terminal (if UTILITY DTYPE is issued from the terminal) or to the program stack (if UTILITY DTYPE is called by another EXEC). If the DASD type cannot be determined using DIAGNOSE code X'24', the same search is performed using DIAGNOSE code X'210'

Only VM/XA-installation-supported DASD types are represented in $DASD$ CONSTS. The information returned is a single line consisting of three fields:

- DASD type and model
- The proper representation of the given DASD model to the CPFORMAT program.

If the virtual device address is invalid, nonexistent, or addresses an unsupported device type, a nonzero return code is displayed.

The UTILITY DTYPE function is intended primarily for use by installation-related utilities.

Usage Notes

1. If you specify ALL for the GEN option, UTILITY generates all of the utilities listed under the GEN option, except CPFMTXA MODULE and HCPSADMP. If you specify ALL for the DVFGEND option, UTILITY generates all of the utilities listed under the DVFGEND option.
2. UTILITY creates HCPLDR MODULE and LOADER by assembling the associated files. It generates all other modules from the associated text deck. The text deck must be on an accessed disk before UTILITY is run. You must run SETUP EXEC, then VMFSETUP EXEC with the CP ALL parameters, to establish the appropriate minidisk access order.

Messages and Return Codes

DMSWUT002E **File *fn ft* not found [RC=28]**
DMSWUT050E **Parameter missing after *value* [RC=24]**
DMSWUT070E **Invalid parameter *parameter* [RC=24]**
DMSWUT095E **Invalid address *vaddr* [RC=24]**
DMSWUT338E **Generating a new *fn ft* will not be attempted [RC=28]**
DMSWUT340E ***file name* has not been created [RC=24]**
DMSWUT971E **Unable to locate *fn ft* [RC=24]**
DMSWUT972I ***fn ft fm* spooled the printer [RC=0]**
DMSWUT973R **Enter the minidisk address where the IPL decks were loaded [RC=0]**
DMSWUT974I **Unable to find IPL decks on the minidisk you indicated [RC=0]**
DMSWUT975I **Moving *fn ft* to tape [RC=0]**
DMSWUT976I **The *fn ft* program is on tape file number *number* [RC=0]**
DMSWUT980I **An IPLable CP nucleus now exists on tape [RC=0]**
DMSWUT986I **Unable to restore ACCESS to *mdisk* [RC=0]**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference.*

VMFAPPLY EXEC

Use the VMFAPPLY EXEC to create auxiliary control files for the PTFs you want to apply to VM/XA System Product.

Format

The format of the VMFAPPLY EXEC is:

VMAPPLY	*prodid* [*compname*] [*updateid*] [(*options...* [)]] *options:* [PUT / COR] [SETUP / NOSETUP] [EXCLUDE / NOEXCLUDE] [CHECK / NOCHECK] [LOG / NOLOG]

Where:

prodid
is the product to be serviced. The *prodid* for VM/XA System Product Release 2.1 is 56643089.

compname
is the component to be serviced. The names for the VM/XA System Product Release 2.1 components are CP, CMS, GCS, and DV.

updateid
is the filetype of the auxiliary control files. It should match a filetype listed in the CNTRL file. If you do not specify an update ID, VMFAPPLY will use the update ID in the product parameter file.

options:

PUT
indicates that you are applying program update service. PUT is the default.

COR
indicates that you are applying COR service.

SETUP
specifies that a new minidisk access order will be set up. SETUP is the default.

NOSETUP
specifies that a new minidisk access order will not be set up. You must set the correct access order up yourself.

EXCLUDE
means that PTFs listed in the exclude list will not be applied. EXCLUDE is the default.

Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prereqisite for the second PTF.

If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

NOEXCLUDE
means that the exclude list will be ignored.

CHECK
means that all the checks that can be specified in the product parameter file will be performed.

NOCHECK
means that none of the checks that can be specified in the product parameter file will be performed.

LOG
specifies that error messages will be written both to the apply exception log and, if critical, to the terminal. LOG is the default.

Notes:

1. Noncritical messages are written only to the apply exception log, not to the terminal.
2. Messages issued by the VMFOVER and VMSETUP EXECS, which VMFAPPLY invokes, are written only to the terminal, not to the apply exception log.

NOLOG
specifies that error messages will be written only to the terminal.

Note: Options specified on the VMFAPPLY command override the options specified in the product parameter file.

How VMFAPPLY Works

Figure 48 on page 647 is an overview of VMFAPPLY processing.

Verifying the Apply List

First, VMFAPPLY checks that the apply list is an appropriate apply list for program update service if you specified PUT, or for corrective service if you specified COR. If it is the wrong apply list, a message is issued.

Creating the Temporary PPF

VMFAPPLY calls VMFOVER to create a copy of the appropriate component section of the product parameter file with overrides. If the product has more than one component and no component was specified, you are shown a list of components and asked to choose one. This copy is used instead of the original product parameter file.

User Exit

A user exit EXEC call for setup and cleanup is provided. The user EXEC is identified in the PPF by the :USEREXIT. tag. If an EXEC name follows the :USEREXIT. tag, that EXEC is called at the beginning and end of the VMFAPPLY EXEC. A separate EXEC can be provided for each component. VMFAPPLY passes parameters indicating the receive function and either setup or cleanup, and a return code from VMFAPPLY. Examples of such calls might be:

```
MYEXEC VMFAPPLY SET-UP
MYEXEC VMFAPPLY CLEAN-UP rc
```

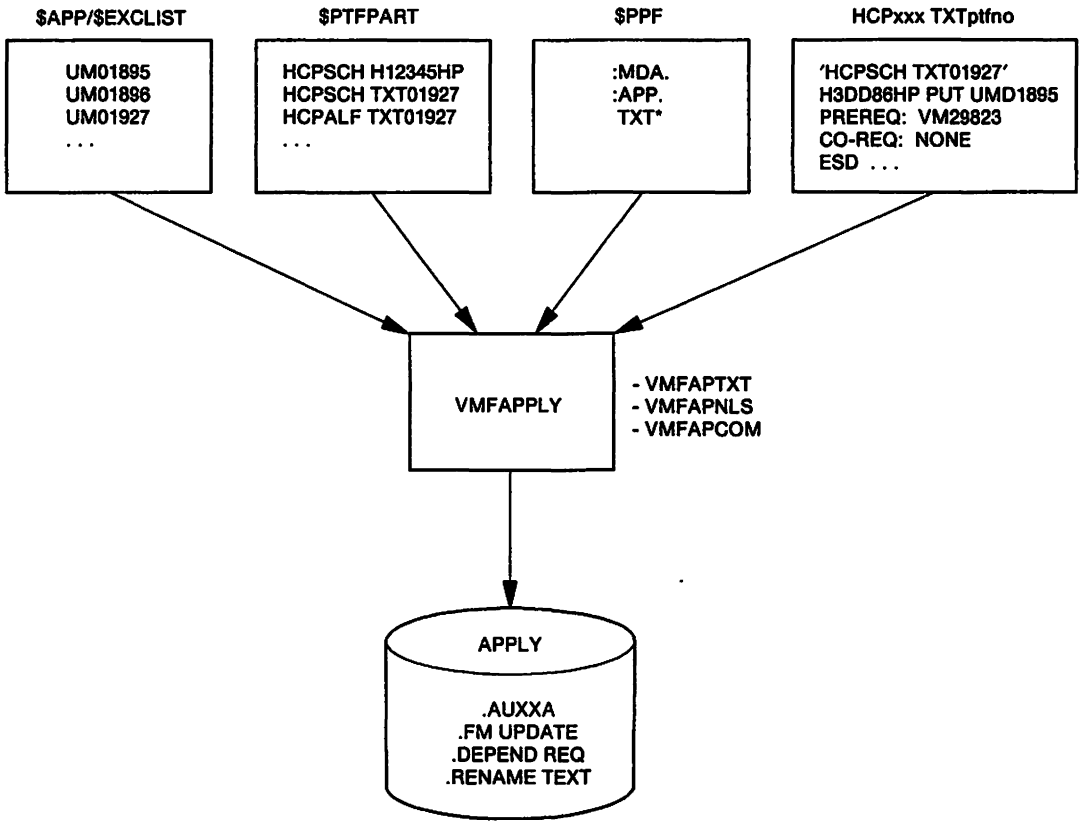


Figure 48. How VMFAPPLY Works

Minidisk Accesses

VMFAPPLY invokes VMFSETUP (unless you specified NOSETUP) to save your current disk access order and build a new disk access order based on the product parameter file. (The product parameter file should be on a system disk accessed in your current search order, preferably your A-disk.) The new search order is:

TASK disks
APPLY disks
DELTA disks
LOCAL disks
BASE disks
BUILD disks
SYSTEM disks.

The PTF Parts List

VMFAPPLY obtains the first PTF entry from the apply list. If the entry is not on the exclude list, VMFAPPLY locates the PTF parts list by using the PTF number (7 characters) for filename and $PTFPART for filetype. VMFAPPLY works through the PTF parts list one at a time and calls the specific part handler based on entries in the PPF part type list.

After the PTF is applied, the filemode of the PTF parts list is changed to filemode 5. The procedure is repeated for each entry in the apply list.

If the :CKUPD tag in the product parameter file is set to NO, and the filemode of the PTF parts list is 5, no processing is performed on the PTF.

Part-Specific EXECs

VMFAPPLY invokes four part-specific EXECs (called part handlers). These part handlers are invoked once per part listed in the PTF parts list. The filetype of the part determines which part handler is used:

1. VMFAPAUX supports AUX files in the apply function.
2. VMFAPTXT supports text parts in the apply function.
3. VMFAPNLS supports text parts that require national language support in the apply function.
4. VMFAPCOM is a common EXEC that insures the existence of parts that do not have specific processing but are listed in the PTF parts list.

VMFAPAUX EXEC: VMFAPAUX is invoked once for each AUX file in each $PTFPART list processed during the execution of VMFAPPLY.

VMFAPAUX locates the cumulative AUX file that contains the given PTF number. If this file does not exist, VMFAPAUX creates it. VMFPAUX validates the AUX file by commenting out duplicates which exist at lower AUX file levels. (These comments are located in the newly created file). If duplicate AUX files exist at higher levels, VMFAPAUX issues a message in the apply exception log.

VMFAPAUX adds included PTFs to the $APPINCL list, adds depend entries to requisite update files, adds requisite update files that don't have a filemode of 5 to the $APPREQ list, and changes the filemode of the update file to 5.

If the file contains new service, VMFAPAUX copies the new AUX file to a file with the filetype listed in the :UPDTID tag of the PPF.

| Applying service to a CMS MACRO or COPY file results in an "M" being inserted after the third character
| of the specified CMS control file.

VMFAPCOM EXEC: VMFAPCOM is called to locate parts named in the PTF parts list for which no other specific processing is provided. Any part not found is listed in the exception log. The filemode is not changed to 5 because this is not a true apply (that is, no AUX structure is built) and there is no corresponding VMFBLD step.

VMFAPTXT EXEC: VMFAPTXT first looks for an existing AUX file (for the specified module and at the specified update level) in the APPLY disk sequence. If the AUX file exists on the current apply disk, it is used as a base and additional entries are added to it. If the existing AUX file is on an earlier disk in the sequence, it is copied onto the current disk and used as a base. If none exists, a new file is created.

VMFAPTXT gets the latest APAR entry (at the bottom of the text deck prolog) and checks the filemode of the update file. If the filemode number is 5, the APAR has already been applied. No further processing is required for that deck. If the filemode is 1, the entry is inserted at the top of the list of AUX file entries and the filemode number is changed to 5. If the update file is not found, it is created and a message is inserted in the apply exception log.

The update files should be found on either the DELTA or the APPLY disks. If they exist on the DELTA disk, source service is available. If they are on the APPLY disk, they are OCO modules with object service only. Changing the filemode from *x*1 to *x*5 indicates that the APAR has been applied (listed in an AUX structure). All applied OCO updates have the filemode of the APPLY disk; and all applied source updates have the filemode of the DELTA disk. All updates that have not been applied, but that do exist on the DELTA or APPLY disk, have the filemode *x*1.

The names (derived from APAR numbers) of update files in which prereq and coreq entries appear are entered in the requisite update files as DEPEND entries if the prerequisites and corequesites are in the same

component as the update being applied. This will facilitate the backout of a bad fix and avoid leaving a dependent fix without its requisites. If the requisite file is not found, an entry is made in the apply exception log. Requisites (*prereq, coreq, ifreq*) that are outside the component will be put in a list to be stored in the apply exception log.

VMFAPTXT proceeds to process all entries (working from bottom to top) in the text deck prolog until an already-applied entry (one whose filemode is 5) is encountered. Each succeeding entry from the text deck is inserted behind the last entry moved from the current text deck, but ahead of any entries that may have come from previous text decks. If the PTF number in any included entry is not in the apply list (minus the exclude list), a message is put in the apply exception log.

If CKTXT is specified in the PPF, and if the top AUX entry is being applied at this time, (that is, not previously applied as indicated by filemode 5 on the update file that was listed in the text deck prolog) VMFAPTXT checks to see if the text deck is a full text deck or a shell only. If it is a shell only, the module name is added to the apply exception log as a missing text deck. If it is a full text deck, VMFAPTXT checks to see if this full text deck supersedes any listed in the apply exception log as missing text decks, in which case the module name is purged from the log. VMFAPTXT reads and writes directly to the log on the user's A-disk.

After all entries have been processed, a search is made for AUX files at higher control file entries. AUX files at these CNTRL file entries may represent local modifications or local text patches. The AUX file filename and filetype are listed in the apply exception log along with all entries from the AUX file.

Finally, VMFAPTXT searches the build lists in the BLD section of the PPF for the text deck filename. If the filename of the text deck being processed is found, a message is be put in the $VMFAPP $ERRLOG indicating what build list the text deck was found in. This allows the user to easily determine what parts can be built using VMFBLD.

After processing is complete, control returns to VMFAPPLY.

If the name is not found in the list, the deck is copied with the rename filetype from the third column of the PPF entry in the apply section to allow compatibility with current MODULE build support. If a deck with the rename filetype already exists, the self-documenting prologs are compared to determine the superset deck that should contain all entries from the other deck. The superset deck is kept with the renamed filetype either by retaining the existing renamed deck or by copying the new deck with the REPLACE option.

After the copy, VMFAPTXT returns to VMFAPPLY.

VMFAPNLS EXEC: VMFAPNLS first creates an AUX filename by concatenating the text filename with a 1- or 2-character NLS suffix based on the text filetype prefix. If the filetype prefix is TAM, the language is mixed-case American English and the suffix is null. All other NLS filename suffixes are based on the file VMFNLS LANGLIST S, which is resident on MAINT's 190 disk and shown in Figure 49.

```
*   AMENG  TAM  American English
A   KANJI  TKA  Kanji
B   UCENG  TUC  Uppercase English
C   PORTG  TPO  Brazilian Portuguese
D   FRANC  TFR  French
E   GER    TGR  German
```

Figure 49. VMFNLS LANGLIST File

VMFAPNLS then looks for an existing AUX file (with the AUX filename created in the previous step and the specified update identifier) in the APPLY disk sequence. If the AUX file exists on the current APPLY

disk, it is used as a base and additional entries are added to it. If the existing AUX file is on an earlier disk in the sequence, it is copied onto the current disk and used as a base. If none exists, a new file is created.

VMFAPNLS gets the latest APAR entry (at the bottom of the text deck prolog) and checks the filemode of the update file. If the filemode number is 5, the update has already been applied. No further processing is required for that deck. If the filemode is 1, the entry is inserted at the top of the list of AUX file entries and the filemode number is changed to 5. If the update file is not found, it is created and a message is inserted in the apply exception log.

The update files should be found on either the DELTA or the APPLY disks. If they exist on the DELTA disk, source service is available. If they are on the APPLY disk, they are OCO modules with object service only. Changing the filemode from *x*1 to *x*5 indicates that the APAR has been applied (listed in an AUX structure). All applied OCO modules have the filemode of the APPLY disk; and all applied source modules have the filemode of the DELTA disk. All modules that have not been applied, but that do exist on the DELTA or APPLY disk, have the filemode *x*1.

The names (derived from APAR numbers) of update files in which prereq and coreq entries appear are entered in the requisite update files as DEPEND entries if the prerequisites and corequesites are in the same component as the update being applied. This will facilitate the the backout of a bad fix and avoid leaving a dependent fix without its requisites. If the requisite file is not found, an entry is made in the apply exception log. Requisites (*prereq, coreq, ifreq*) that are outside the component will be put in a list to be stored in the apply exception log.

Messages indicating a requisite outside the component have a prefix of RO. These are not error messages, but informational messages issued so that you are sure to find all the requisites.

VMFAPNLS proceeds to process all entries (working from bottom to top) in the text deck prolog until an already-applied entry (one whose filemode is 5) is encountered. Each succeeding entry from the text deck is inserted behind the last entry moved from the current text deck, but ahead of any entries that may have come from previous text decks. If the PTF number in any included entry is not in the apply list (minus the exclude list), a message is put in the apply exception log, and the PTF is added to a supplemental apply list (called *fn* $APPINCL) to be processed after the main apply list.

If CKTXT is specified in the PPF, and if the top AUX entry is being applied at this time, (that is, not previously applied as indicated by filemode 5 on the update file that was listed in the text deck prolog) VMFAPNLS checks to see if the text deck is a full text deck or a shell only. If it is a shell only, the module name is added to the apply exception log as a missing text deck. If it is a full text deck, VMFAPNLS checks to see if this full text deck supersedes any listed in the apply exception log as missing text decks, in which case the module name is purged from the log. VMFAPNLS reads and writes directly to the log on the user's A-disk.

After all entries have been processed, a search is made for AUX files at higher control file entries. AUX files at these CNTRL file entries may represent local modifications or local text patches. The AUX file filename and filetype are listed in the apply exception log along with all entries from the AUX file.

Finally, VMFAPNLS copies the deck with the NLS filetype indicated in the PPF. If a deck with the NLS filetype already exists, the self-documenting prologs are compared to determine the superset deck that should contain all entries from the other deck. The superset deck is kept with the NLS filetype either by retaining the existing NLS text deck or by copying the new deck with the REPLACE option.

After the copy, VMFAPNLS returns to VMFAPPLY.

Return Codes

VMFAPPLY issues the following return codes:

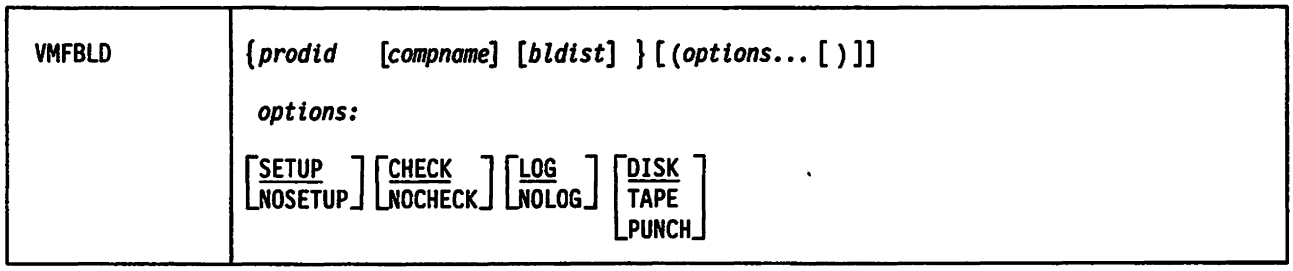
Return Code	Explanation
0	No errors were encountered.
4	A message of type CK, WN, RO, or BD was encountered. Check $VMFAPP $ERRLOG.
8	There was syntax error in the command.
24	There was an unexpected tag in PPF.
28	The required file was not found.
100	An unrecoverable error was encountered.

VMFBLD EXEC

Use the VMFBLD EXEC to build the nucleus during installation and to rebuild the nucleus after applying service. VMFBLD invokes $VMFPAT$, which invokes the HCPLDR command.

Format

The format of the VMFBLD EXEC is:



Where:

prodid
is the product to be serviced. The *prodid* for VM/XA System Product Release 2.1 is 56643089.

compname
is the component whose nucleus is to be rebuilt.

The names for the VM/XA System Product Release 2.1 components are CP, CMS, GCS, and DV.

bldlist
is the loadlist or buildlist for the nucleus or module you wish to rebuild. If you do not specify *bldlist*, all entries for the component will be processed.

options:

SETUP
specifies that a new minidisk access order will be set up.

NOSETUP
specifies that a new minidisk access order will not be set up. You must set the correct access order up yourself.

CHECK
means that all the checks that can be specified in the product parameter file will be performed.

NOCHECK
means that none of the checks that can be specified in the product parameter file will be performed.

LOG
specifies that error messages will be written both to the build exception log and, if critical, to the terminal.

Notes:

1. Noncritical messages are written only to the build exception log, not to the terminal.
2. Messages issued by the VMFOVER and VMSETUP EXECs, which VMFBLD invokes, are written only to the terminal, not to the build exception log.

NOLOG
specifies that error messages will be written only to the terminal.

DISK
specifies that the nucleus will be written to the target disk specified in the product parameter file. This must be the same disk specified on the SYSRES macro in the HCPSYS file (for CP) or the DMSNGP profile (for CMS).

TAPE
specifies that the nucleus will be written to tape 0182 in card image format.

PUNCH
specifies that the nucleus will be spooled to your virtual card reader.

Note: Options specified on the VMFBLD command override the options specified in the product parameter file.

How VMFBLD Works

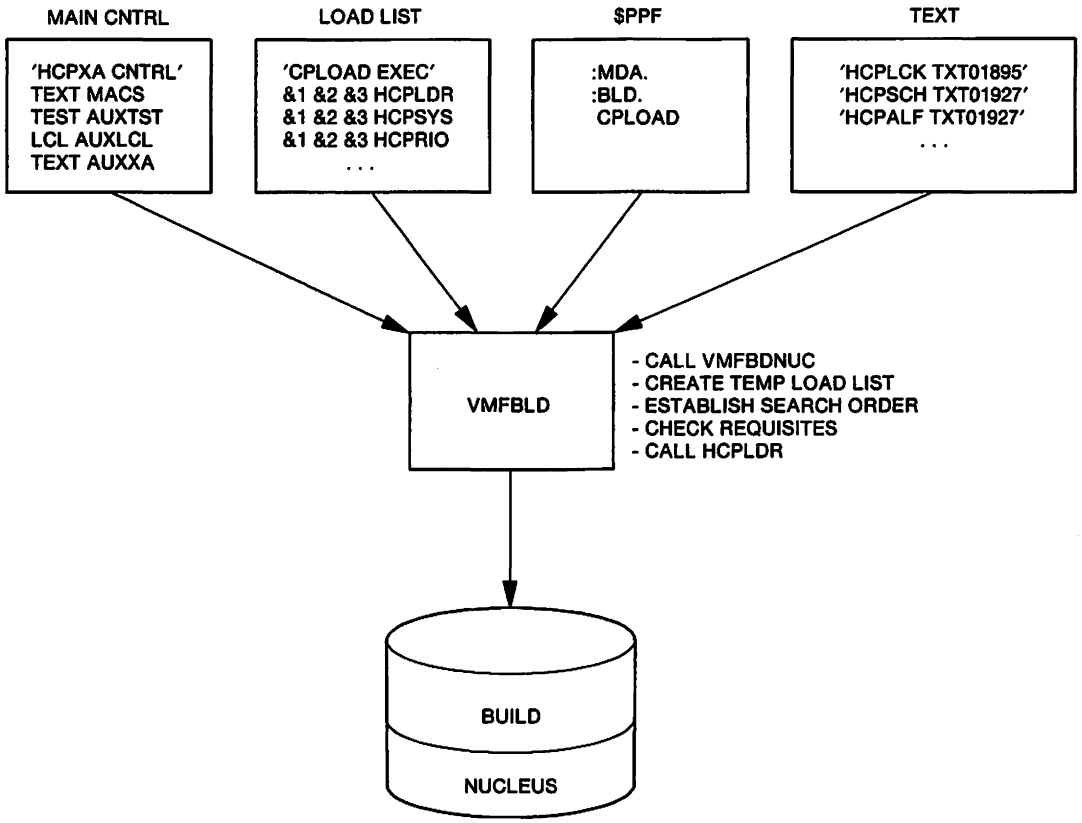


Figure 50 (Part 1 of 2). How VMFBLD Works

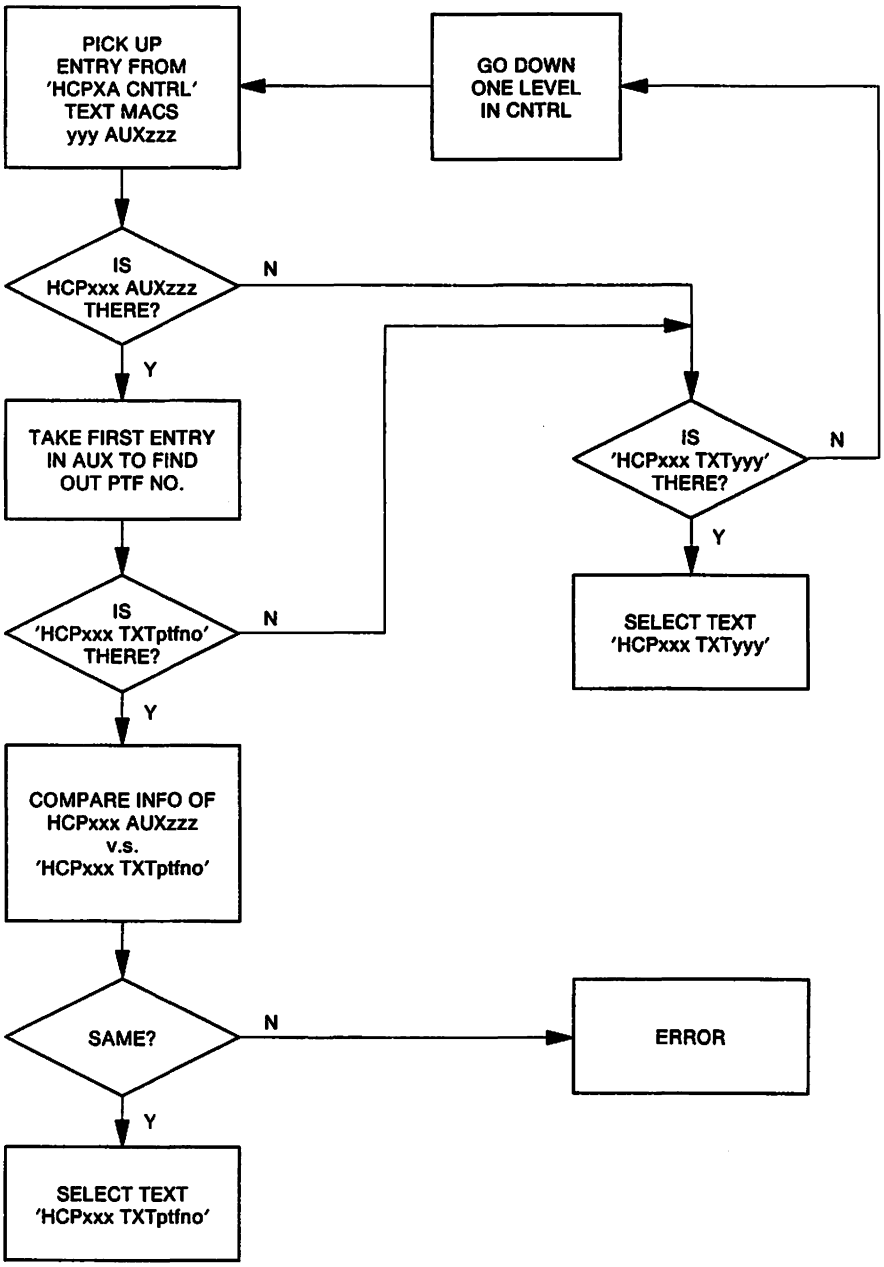


Figure 50 (Part 2 of 2). How VMFBLD Works

Creating the Temporary PPF

VMFBLD calls VMFOVER to create a copy of the appropriate component section of the product parameter file with overrides. This copy is used instead of the original product parameter file.

User Exit

A user exit EXEC call for setup and cleanup is provided. The user EXEC is identified in the PPF by the :USEREXIT. tag. If an EXEC name follows the :USEREXIT. tag, that EXEC is called at the beginning and end of the VMFBLD EXEC. A separate EXEC can be provided for each component. VMFBLD passes parameters indicating the receive function and either setup or cleanup, and a return code from VMFBLD. Examples of such calls might be:

```
MYEXEC VMFBLD SET-UP
MYEXEC VMFBLD CLEAN-UP rc
```

Minidisk Accesses

VMFBLD then invokes VMFSETUP (unless you specified NOSETUP) to save your current access list and establish a new list based on the product parameter file. The new access order is:

TASK disks
LOCAL disks
APPLY disks
DELTA disks
BASE disks
BUILD disks
SYSTEM disks.

Part-Specific EXECs

VMFBLD invokes the following part-specific EXECs:

- VMFBDNUC creates a temporary load list and invokes the system loader.
- VMFBDCPY copies the PTF-numbered TEXT deck to the BUILD disk.

VMFBDNUC EXEC: VMFBDNUC creates a new (temporary) load list named $$$TLL$$ EXEC that contains both filename and filetype to pass to HCPLDR. VMFBDNUC will call the VMFLDS MODULE to perform a search for the TEXT decks to load and create the temporary load list.

When the temporary load list has been completed, the list of requisites within the component is processed as follows if the CHKREQ option has been specified. The list of requisite APARs is created by the VMFLDS MODULE.

1. VMFBDNUC creates actual update filetypes by combining the numeric portion of the APAR number with the system and version level from the PPF. For example:

APAR number	VM12345
System and version (and alternate)	S/DK A/XX
Numeric portion of APAR number	12345
Possible update filetypes	S12345DK A12345XX

2. VMFBDNUC creates a list of update files by filename and filetype. For example:

```
listfile * S12345DK * ■
MODABC S12345DK
MODXYZ S12345DK
listfile * A12345XX * ■
MODDEF A12345XX
```

3. VMFBDNUC locates MODABC, MODXYZ and MODDEF in the temporary load list and obtains the filetype (PTF level) of the text deck included in the nucleus build.
4. VMFBDNUC searches the self-documenting prolog of the text deck to be sure that the APAR number (in this example, 12345) is included. If not, a message is put in the build exception log.

After requisite checking has completed, VMFBDNUC calls the $VMFPAT$ EXEC with the new load list containing filenames and filetypes of text to be loaded. $VMFPAT$ looks for patches that need to be applied to text decks in the load list. Any patches found are inserted into the appropriate text deck and included in the IPL deck produced by the loader.

When the loader has completed punching the text decks, control returns to VMFBLD.

VMFBDCPY EXEC: VMFBDCPY will be used to copy the PTF-numbered TEXT deck to the BUILD disk identified in the PPF. VMFBDCPY calls VMFLDS to create a new (temporary) load list named $$$TLL$$ EXEC, which contains both the filenames and filetypes of text decks to load. Requisite checking is performed as in VMFBDNUC. VMFBDCPY then invokes the COPYFILE command to copy the PTF-numbered deck to the BUILD disk and to rename it. These files must reside on the BUILD disk with a filetype of TEXT.

Return Codes

VMFBLD issues the following return codes:

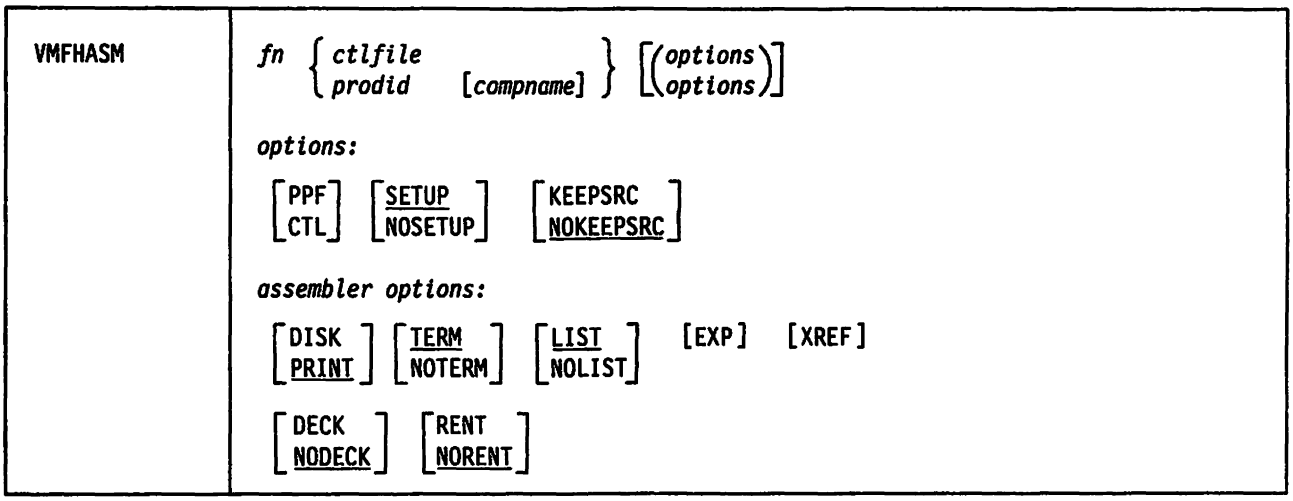
Return Code	Explanation
0	No errors were encountered.
4	A message of type CK, WN, RO, or BD was encountered. Check VMFBLD $ERRLOG.
8	There was a syntax error in the command.
24	There was an unexpected tag in PPF.
28	The required file was not found.
100	An unrecoverable error was encountered.

VMFHASM EXEC

The VMFHASM EXEC updates a specified source file according to entries in a control file, and assembles the updated source file. VMFHASM invokes the CMS UPDATE command, then assembles the updated source file using Assembler H Version 2. If you wish, VMFHASM can use the product parameter file to establish the minidisk search order.

Format

The format of the VMFHASM EXEC is:



Where:

fn
is the filename of the source file to be updated. It must have a filetype of ASSEMBLE.

ctlfile
is the filename of the control file. The control file must have a filetype of CNTRL. If a product parameter file exists with a filename of *prodid*, the name of the CNTRL file will be found in the product parameter file under "Compname." You do not have to specify the *ctlfile* parameter.

prodid
is the product identifier. The *prodid* for VM/XA System Product Release 2.1 is 56643089.

compname
is the component name specified in the product parameter file. The names for the VM/XA System Product Release 2.1 components are CP, CMS, GCS, and DV.

options:

PPF
indicates that the second parameter is a product parameter filename. If neither CTL nor PPF is specified, the VMFHASM EXEC searches for a file whose name is the same as the second parameter. If such a file is found, its filetype determines whether it is a control file or a product parameter file. If both exist, the product parameter file is used.

CTL
indicates that the second parameter is a control filename.

SETUP
specifies that a new minidisk access order will be set up.

NOSETUP
specifies that a new minidisk access order will not be set up. You must set the correct access order up yourself.

KEEPSRC
indicates that the merged source file consisting of the updates and the base source file will be saved on the user's disk. The file is named *$filename filetype*, where *filename* is the original source input filename and *filetype* is the original source input filetype.

NOKEEPSRC
does not save the merged source file.

***assembler options*:**

DISK
places the listing file on a virtual disk.

PRINT
writes the listing file to the printer.

TERM
writes the diagnostic information on your display. The diagnostic information consists of the diagnosed statement followed by the error message issued.

NOTERM
suppresses the TERM option.

LIST
produces an assembler listing.

NOLIST
does not produce an assembler listing.

DECK
creates an object module.

NODECK
suppresses the DECK option.

RENT
checks the program for a possible violation of program reenterability. An error message identifies code that makes the program nonreenterable.

NORENT
suppresses the RENT option.

EXP
prints all macro expansions. By default, macro expansions are not printed.

XREF
causes the XREF(FULL) option to be invoked when VMFHASM invokes the assembler. The default for VMFHASM is XREF(SHORT).

How VMFHASM Works

The steps taken by the VMFHASM EXEC are summarized below:

1. If a product parameter file is used, VMFHASM calls VMFOVER to create a copy of the appropriate component section of the product parameter file with overrides. This copy is used instead of the original product parameter file.

VMFHASM then invokes VMFSETUP (unless you specified NOSETUP) to save your current disk access order and build a new disk access order based on the product parameter file.

2. The VMFHASM EXEC calls the UPDATE command, which includes the CTL, STK, and PRINT options.

   UPDATE uses the control file (*ctlfile* CNTRL) to update the assembler language source file. The new file is named *$fn* ASSEMBLE.

   UPDATE stacks information from the control file in the console stack, and prints the update log file.

3. Using the library list from the MACS record in the control file, VMFHASM issues a GLOBAL MACLIB command.

4. The updated source file, *$fn* ASSEMBLE, is assembled using the options indicated on the VMFHASM command line.

5. The output text deck from the assembly, *$fn* TEXT, is concatenated with the UPDATES file so that the text deck contains a history of update activity.

6. UPDATE stacks the identifier of the most recent update it found and applied. (If no updates were found, UPDATE stacks the identifier of the MACS record in the main control file instead.) If no PTF number or local tracking number is available, VMFHASM uses this update level identifier to determine how to rename *$fn* TEXT.

   If a product parameter file was specified (or used by default), and if the update was listed in an auxiliary control file containing the PTF number or local tracking number, the text deck is renamed *fn xxxnnnnn*, where *xxx* is the text filetype prefix from the corresponding record in the main control file and *nnnnn* is the PTF number or local tracking number. The default textfile type prefix is TXT.

   If a control file was specified (or used by default); or if the PTF number or local tracking number is not available, and if the update level identifier is TEXT, the text deck is renamed *fn* TEXT. If the update level identifier is anything other than TEXT, the text deck is renamed *fn* TXT*xxxxx* (where *xxxxx* is the 1- to 5-character update level identifier).

7. Temporary files (*$fn* ASSEMBLE and *fn* UPDATES) are erased.

The input and output files used by VMFHASM are summarized below:

Input and Output Files

Disk Input Files

fn ASSEMBLE	Assembler language source
fn CNTRL	Control file
fn MACLIB	Macro library
fn AUX*xxxxx*	Auxiliary control files
fn UPDT*xxxx*	Update files
fn $PPF	Product parameter file.

Disk Output Files

fn {TEXT|TXT*xxxxx*}	Object deck, named according to the update level identifier in the control file. This file also contains data from the UPDATES file, together with date and time information. It is placed on the A-disk, which you must access in read/write mode before issuing VMFHASM.

Printer Output Files

\$*fn* LISTING Assembler listing (if PRINT option is in effect). This file also contains data from the update log file (*fn* UPDLOG), describing the updates applied to the source file.

Messages

The UPDATE command issues messages to indicate when each update file is applied.

ASMBLING *fn*

Explanation: The assembly is going to begin. If you specified any assembler option on the VMFHASM command line, the options used are also displayed.

***fn* {TEXT|TXT*xxxxx*} CREATED**

Explanation: This message indicates the filename and filetype of the text deck.

Note: If the source is on some other R/W minidisk instead of the A minidisk, VMFHASM will not create a text deck identifying the update level on the A minidisk.

VMFLKED EXEC

The VMFLKED EXEC procedure invokes the CMS LKED command to link-edit modules into a LOADLIB. VMFLKED uses the normal CMS search order when searching for TEXT files.

Format

The format of the VMFLKED EXEC is:

VMFLKED	*fn* [ft / LKEDCTRL] [fm / *] [(*options...*[)]] *options:* [PRINT] [MODULE *module_name*]

Where:

fn
is the filename of the input control file. You must specify a filename.

ft
LKEDCTRL
is the filetype of the input control file. The default filetype is LKEDCTRL.

fm
*
is the filemode of the input control file. The default filemode is *.

options:

PRINT
prints out a hard copy of the linkage-editor output.

MODULE *module_name*
indicates that only those members of the LOADLIB that include *module_name* are to be link-edited.

How VMFLKED Works

VMFLKED reads the specified LKEDCTRL file and expects to find option records (if any are needed) followed by linkage-editor input records. Multiple groups of options followed by linkage-editor input can be combined in a single LKEDCTRL file (see Figure 51 on page 664).

1. VMFLKED processes the option records, which are identified by a percent sign (%) in column one.
2. VMFLKED processes linkage-editor input records when it finds a nonoption and noncommentary record.

   When VMFLKED finds:

   - An INCLUDE record, the process adds the record to the linkage-editor input file and issues a FILEDEF for the TEXT file
   - A NAME record, the process adds the record to the linkage-editor input file and invokes the linkage editor

- A commentary record (beginning with *), the process ignores the record
- Any other record, the process adds the record to the linkage-editor input file

3. VMFLKED continues with Step 1 to process the next group of option records (if any).

Input and Output Files

Input Files

fn TEXT — The name of the TEXT file. When the process finds an INCLUDE statement in the linkage-editor input control file, it issues a FILEDEF for that TEXT file. The linkage editor reads this TEXT file.

fn LKEDCTRL — A modified linkage-editor file. INCLUDE cards include TEXT files, and you cannot use them to include files from a library. Any record containing an asterisk (*) in column one is commentary and the process ignores it.

The VMFLKED EXEC also recognizes special control (option) records in the linkage-editor control file (see Table 19). These records, which must begin with a percent sign (%) in column one, may only be located between linkage-editor input files (that is, groups of them may be at the beginning of the file or following a NAME record).

Table 19 (Page 1 of 2). Linkage-Editor Control File Special Control (Option) Records

Record	Parameter	Function
%CONTROL	*control_file_name*	This record indicates to take the filetypes for linkage-editor input files from a control file. The name of the control file is specified on the %CONTROL record and the filetype is CNTRL. When the %CONTROL record is read, VMFLKED reads the control file and uses the first "word" of each line of the file to build an array of filetypes. The first line of the file indicates the default filetype. Each of the filetypes is checked and if it is not TEXT, the characters TXT are put at the beginning. When an VMFLKED finds an INCLUDE card, it searches the array. VMFLKED uses the first filetype from the array that matches an existing file. This option is in effect until a %NOCONTROL record is found.
%NOCONTROL		This record indicates not to take the filetype for linkage-editor input files from a control file.
%LIBRARY	*load_library_name*	This record changes the LOADLIB to be used and the name on the LKEDIT listing. The default for the LOADLIB name is the filename of the LKEDCTRL file.
%LEPARMS	*link-edit parameters*	This record sets the link-edit parameters used in each link-edit step. If no parameters are specified, none are used.

Table 19 (Page 2 of 2). Linkage-Editor Control File Special Control (Option) Records

Record	Parameter	Function
%MAXRC	*maximum_valid_return_code*	This record sets the maximum valid return code. The VMFLKED EXEC checks this value when it ends. If the highest return code is higher than this value and is not listed on the %ACCEPTRC record, the EXEC issues a warning message.
%ACCEPTRC	*return codes*	This record lists the acceptable return codes for VMFLKED processing. (The return codes are usually higher than entry on the %MAXRC entry). VMFLKED checks the values after each link-edit. If the return codes after any link-edit are higher than the %MAXRC record and not specified on the $ACCEPTRC record, VMFLKED issues a warning message at the end of its processing.
%IGNORE		This record causes the EXEC to bypass the warning message for missing TEXT files. Once specified, this record takes effect for all subsequent link-edits (or until a %NOIGNORE record is found).
%NOIGNORE		This record causes a warning message to be issued if there is a missing TEXT file. Once specified, this record takes effect for all subsequent link-edits (or until a %IGNORE record is found).
%ERASE		This record erases LOADLIB and LKEDIT files. This is useful when you rebuild the LOADLIB as it keeps the LOADLIB and LKEDIT files as small as possible. If this record is not specified, the LOADLIB and LKEDIT files are not erased.

Note: The %ERASE control statement takes effect immediately, and erases whatever is the current LOADLIB. The LOADLIB is not erased if you use the MODULE option. (The LOADLIB can be changed using the %LIBRARY control statement).

Output Files

fn LOADLIB — The main output from the linkage editor. This file contains the link-edited load modules.

fn LKEDIT — The file that contains the linkage-editor map for all modules.

The following is an example of a LKEDCTRL file.

```
%CONTROL YOURCTRL
%LIBRARY NCCF
%ERASE
%MAXRC 4
%ACCEPTRC 12 14
%LEPARMS NCAL LIST XREF LET RENT
  INCLUDE  DSIZDST
  INCLUDE  DSIZSHP
  INCLUDE  DSILUTRM
  ORDER    DSIZDST
  ENTRY    DSIZDST
  NAME     DSIZDST(R)
%LEPARMS NCAL LIST XREF LET REUS
  INCLUDE  DSIVMCMD
  INCLUDE  DSIVMMSG
  ORDER    DSIVMCMD
  ENTRY    DSIVMCMD
  NAME     DSIVMCMD(R)
%IGNORE
%LEPARMS NCAL LIST XREF LET RENT
  INCLUDE  DSIPRTVM
  ENTRY    DSIPRTVM
  NAME     DSIPRTVM(R)
```

Figure 51. Example of a LKEDCTRL file

Messages

DMSWLK002E **File *fn ft* [*fm*] not found**
DMSWLK002W **File *fn ft* not found**
DMSWLK003E **Invalid option *opt***
DMSWLK005E **No {LKEDCTRL|MODULE} specified**
DMSWLK010E **Premature EOF on file *fn ft***
DMSWLK065E **{PRINT|MODULE} option specified twice**
DMSWLK842E **No {control|library} filename found in *fn ft* [*fm*]**
DMSWLK843I **An invalid control record was found and ignored:**
DMSWLK844E **No linkedit performed**
DMSWLK845W **Errors were encountered during the link edit processing that will probably make the loadlib unusable**
DMSWLK846I **LKED *target_module* into *library***

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference.*

VMFMAC EXEC

The VMFMAC EXEC updates macro libraries. It invokes the CMS UPDATE command to update specified copy[1] or macro files, according to entries in a control file, and then builds a new macro library from the resulting new versions of those files.

Format

The format of the VMFAC EXEC is:

VMFMAC	libname [*ctlfile*]

Where:

libname
is the filename of the macro library to be updated, and of the EXEC file that contains the names of the library members. The entries in *libname* EXEC must be in the following format:

&1 &2 *fn1*
&1 &2 *fn2*

⋮

where *fn1*, *fn2*, and so on, are filenames of macro or copy files to be updated and included in the macro library, which must have a filetype of MACLIB.

ctlfile
is the filename of a control file to be used to apply the updates. The filetype must be CNTRL.

How VMFMAC Works

1. VMFMAC locates *libname* EXEC and the control file. It also erases any existing files named NEWMAC MACLIB and NEWMAC COPY. VMFMAC begins reading the macro or copy filenames from the EXEC, beginning at the bottom.
2. For each entry in the *libname* EXEC, VMFMAC:
   - Invokes the UPDATE command with its CTL option to apply the updates specified in the control file
   - Adds the updated macro or copy file (*$file name* MACRO or *$filename* COPY) to the macro library NEWMAC MACLIB
   - Adds the UPDATES file created by the UPDATE command to the file NEWMAC COPY
   - Erases *$filename* MACRO or *$filename* COPY, and filename UPDATES

[1] Copy files are collections of assembler statements that many modules use in common. In source listings, the assembler includes copy files by a "COPY *copyfilename*" statement. They are similar to macro definitions because they reside in libraries, but they contain no substitution data. VM/XA System Product uses copy files to define control block DSECTS, data areas, and tables.

3. If there are no update files corresponding to a macro or copy file entry specified in *libname* EXEC, the macro or copy file is added to NEWMAC MACLIB in its current form. NEWMAC COPY, containing a history of updates applied by VMFMAC, is added to NEWMAC MACLIB.
4. If no errors occur during the procedure, and when all the macros have been added to NEWMAC MACLIB, erase *libname* MACLIB, if it exists, and rename NEWMAC MACLIB to *libname* MACLIB.

   If errors occur during the VMFMAC EXEC procedure (for example, if a MACRO or a COPY file is not found), *libname* MACLIB is not erased, and the updated macro library retains the name NEWMAC MACLIB.

Input and Output Files

Disk Input Files

libname EXEC	List of macro or copy files to be updated or included (or both) in *libname* MACLIB
ctlfile CNTRL	Control file used by the UPDATE command
fn MACRO	Files to be updated or included (or both) in the macro library.
fn COPY	
⋮	
Auxiliary control files	
Update files	

Disk Output Files

libname MACLIB	Updated macro library
libname COPY	Contains the UPDATE files produced by UPDATE command processing.

Printer Output Files

The printer is spooled with the CONT option so that when VMFMAC completes, the printer file contains:

- A copy of the control files
- The update log file for each updated macro or copy file the UPDATE command produced
- A copy of each macro or copy file in the macro library
- The *libname* COPY file, which contains the accumulated UPDATES files the UPDATE command created.

Notes:

1. When VMFMAC adds files with MACRO filetypes to a MACLIB, the EXEC takes the membername from the macro prototype statement. When VMFMAC adds files with COPY filetypes to a MACLIB, the EXEC follows these rules:
   a. If the membername is from the COPY file and updates were found, the membername is *$filename*.
   b. If the membername is from the COPY file and no updates were found, the membername is *filename*.
   c. If you include a *COPY statement as the first record in the file, the membername is that designated on the *COPY statement. The format of the *COPY statement is:

      ```
      *COPY membername
      ```

2. If errors occur during VMFMAC processing, consult the NEWMAC COPY file printed by VMFMAC. If you can correct the errors involving one or two macro or copy files, you can use the MACLIB command to add these members to NEWMAC MACLIB. Then erase the current *libname* MACLIB and rename NEWMAC MACLIB to *libname* MACLIB.
3. If any accessed disk contains a MACRO file and a COPY file with the same name, the MACRO version is used.

Messages

The UPDATE command issues the message DMSUPD178I to inform you of the updates being applied to each macro or copy file. If no updates are found, message DMSUPD181E is issued.

fn **{COPY|MACRO} ADDED**

Explanation: The specified macro or copy file has been added to the macro library.

libname **COPY ADDED**

Explanation: *libname* COPY, containing the update history of the MACLIB, has been added to the macro library.

VMFMERGE EXEC

The VMFMERGE EXEC procedure applies PTFs (program temporary fixes) from the DELTA disk to the MERGE disk.

Do not use this procedure to service any of the base components of VM/XA SP. Use this procedure when applying PTFs to System Network Architecture (SNA) products.

VMFMERGE requires a service control file (*ptfnum* SCF) for each requested PTF and its requisites. The service control file contains instructions for applying the PTF. To use VMFMERGE, you must access the minidisk containing the *prodid* VMFPARM file. This file identifies the minidisks that VMFMERGE must access to service each product.

Format

The format of the VMFMERGE EXEC is:

```
VMFMERGE   prodid  { PTF [ptfnum|*] }  { HIST   }  [EXCLUDE exclist]
                   { PTFLIST applist }  { NOHIST }
```

Where:

prodid
is the ID of the product you specify.

You cannot specify a *prodid* of EXCLUDE, because EXCLUDE is a keyword for this EXEC.

PTF [*ptfnum*|*]
applies a single PTF if you specify a PTF filename (*ptfnum*). If you enter an asterisk (*) instead of a PTF filename, you will apply all PTFs for the product, as listed in the apply list, called *prodid* APPLIST, supplied on the service tape. You cannot specify a filename of EXCLUDE, because EXCLUDE is a keyword for this EXEC.

If the service tape contains an exclude list file named *prodid* EXCLIST, any PTF listed in that file is not applied.

PTFLIST ***applist***
applies the selected PTFs listed in the apply list file, *applist* APPLIST. If you specify a filename of EXCLUDE, you cannot use the EXCLUDE option to specify an Exclude List.

If the service tape contains an exclude list file named *prodid* EXCLIST, any PTF listed in that file is not applied.

HIST
includes **apartext** comments from the SCF files in the text decks processed. The **apartext** lines from the SCF files are included at the beginning of each text deck as comments. This option is effective for the duration of the command. All text decks processed during the VMFMERGE process will include **apartext** entries as comments.

NOHIST
does not include **apartext** lines from the SCF files as comments in the text decks processed. This option is effective unless specifically overridden with the HIST option.

EXCLUDE *exclist*

excludes the selected PTFs listed in the exclude list file, *exclist* EXCLIST. PTFs listed in *prodid* EXCLIST are excluded, in addition to those in the user-specified exclude list. If you specify EXCLUDE as the apply list filename, you cannot use the EXCLUDE option to specify an exclude list.

If you use *prodid* EXCLIST as the name of your exclude list, then VMFMERGE ignores any other *prodid* EXCLIST file, including the one that may be supplied on the service tape, during processing. You should therefore use some other name for the exclude list you create.

Note: If two PTFs are in the same text deck, you cannot exclude the first unless you also exclude the second, even if you list the first one on the exclude list. This restriction applies because the first PTF is considered to be a prereqisite for the second PTF.

If you list the first PTF in the exclude list, but not the second, and if the second PTF appears in the apply list, both PTFs will be applied. You will receive an error message indicating that a PTF you wanted to exclude has been applied.

How VMFMERGE Works

VMFMERGE is a service process that processes PTFs. The EXEC procedure:

1. Uses the parameter file (*prodid* VMFPARM) to determine the virtual address of the merge and delta disks.
2. Checks the merge log to insure that the PTF you select is not already merged or superseded.
3. Reads the service control file to get the prerequisite and corequisite PTFs and the elements affected by this PTF.
   - If the service control file for any of the prerequisite or corequisite PTFs is missing, processing for the current PTF stops.
   - If a prerequisite or corequisite PTF is in the exclude list, the current PTF is not merged.
   - If a prerequisite or corequisite PTF has been superseded, that prerequisite or corequisite is not merged.
   - If the requisite is not within this product, the system displays a message indicating that the requisite PTF must be merged at some later time.
   - When you merge a PTF that is a requisite of a change in another product, be sure to note this requisite information. There is no automatic way to tell you of this cross-product requisite if at a later time you remove the change that is a requisite of a change in another product.
4. Does the necessary COPY/RENAME from the delta disk to the merge disk for each element in the prerequisite and corequisite chain that was not superseded or already merged.

   Note: Temporary files are created during this COPY/RENAME process to insure system integrity.

   These files are erased during normal VMFMERGE processing.
5. Adds service history to the element if it is a text deck. Element history consists of text deck comments containing:
   - The PTF or APAR number
   - A time and date stamp
   - Any **apartext** information that was in the service control file SCF.

   Note: A temporary file ($APARTXT $VMFMERG) is created during this history process to ensure system integrity. VMFMERGE erases this file during normal processing.
6. Updates the **reqby** log to reflect all the requisite relationships of all the merged and superseded PTFs.

7. Marks in the merge log any ZAPs and PTFs that are superseded by this PTF. Those ZAPs and PTFs are never applied.
8. Puts entries in the merge log to show which PTFs have been merged.

Notes:

1. You can issue VMFMERGE to process a single PTF, a list of PTFs, or all PTFs on the input disk for a product.
2. Merged PTFs cannot be excluded, but can be superseded.
3. Superseded PTFs cannot be merged nor excluded.
4. PTFs in the Exclude List can be superseded.
5. If the following conditions are both true:
   a. VMFMERGE has the HIST option specified, and
   b. LKED does not have the LET option specified,

   error message IEW0222 will not occur and there will not be linkage editor errors due to comments (invalid card).

Messages

DMSWMG002E **File *fn ft* [*fm*] not found.**
DMSWMG008E **Device *vdevno* invalid or nonexistent.**
DMSWMG017E **Invalid device address *vdevno*.**
DMSWMG520E **Invalid operand: *operand*.**
DMSWMG545E **Missing operands.**
DMSWMG649E **Extraneous parameter *parm*.**
DMSWMG653E **Error executing *command*.**
DMSWMG856E **Disk address *vdevno* is listed more than once on the DELTA and/or MERGE entry records in the *prodid* VMFPARM file.**
DMSWMG857E **The number of disk addresses on the DELTA entry record cannot exceed nine.**
DMSWMG858E **Unable to find a *tag* entry record in the *fn ft* file.**
DMSWMG859E **The *prodid* VMFPARM file has no disk addresses on the {MERGE|DELTA} entry record.**
DMSWMG860E **Only one {MERGE|DELTA} entry record may appear in the *prodid* VMFPARM file.**
DMSWMG861I **Accessing *disk_type* disk *vdevno* as *fm*.**
DMSWMG862I **Change *name* has been *action*.**
DMSWMG863E **The MERGE disk *vdevno* must be linked read-write.**
DMSWMG864E **PTF *name* will not be *action* because it already is *status*.**
DMSWMG864I **PTF *name* will not be *action* because it already is *status*.**
DMSWMG864W **PTF *name* will not be *action* because it already is *status*.**
DMSWMG865I **Processing PTF *name*.**
DMSWMG866W **No PTFs have been *action*.**
DMSWMG867E **Invalid status *status* in *prodid* VMFMGLOG for entry *ptf*.**
DMSWMG868E **PTF *name* is not a part of product *prodid*.**
DMSWMG869E **Error in file *fn ft fm*. *data* is invalid for *tag* tag.**
DMSWMG870E **Error in file *fn ft fm*. There are no elements.**
DMSWMG871E **Error in file *fn ft fm*. The *name* tag is missing.**
DMSWMG872E **Error in file *fn ft fm*. REPLACE tag missing after the element *name*.**
DMSWMG873E **Error in file *fn ft fm*. *parm* is an invalid parameter. Expecting parameter(s) PRODID, PREREQ, COREQ, SUP, APARTEXT, or CHANGES.**
DMSWMG874E **Invalid entry found at line *line* in *fn ft*.**
DMSWMG882W **File *fn ft* [*fm*] [from *name*] not found on any DELTA disks from the VMFPARM file.**
DMSWMG883W **PTF *name* is not a part of product *prodid* and must be *action* in product *prodid*.**

DMSWMG886E **Filename *name* from the *fn ft* [*fm*] file is longer than 8 characters.**
DMSWMG892E **PTF *name* has not been *action*.**
DMSWMG893E **Incomplete processing, not all [required] PTFs were *action*.**
DMSWMG893W **Incomplete processing, not all [required] PTFs were *action*.**
DMSWMG898E **VMFREMOV processing is incomplete.**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference*.

VMFNLS EXEC

The VMFNLS EXEC procedure updates national language-related files. VMFNLS automatically applies updates to source files, generates text files, and renames them so they can be loaded into the system. If you have your own language files, this EXEC is provided so that you can maintain those files.

Format

The format of the VMFNLS EXEC is:

VMFNLS	*fn ft* { *ppfname* [*compname*] / *ctlfile* } [(*options...* [)]] *options:* [PPF / CTL] [SETUP / NOSETUP] [KEEPSRC / NOKEEPSRC]

Where:

fn
is the filename of the source file that is to be converted to text.

ft
is the filetype of the source file that is to be converted to text. Only ASSEMBLE, REPOS, and DLCS filetypes are allowed.

ppfname
is the filename of either the base product parameter file (which is the product identification number, *prodid*) or a product parameter override file containing an override area for the component. The filetype must be $PPF.

compname
is the name of either a base component area (which is a base component name, such as CP or CMS) or a component override area containing alternative or additional parameters (or both) for the component. If you do not specify a valid name, VMFNLS shows you a list of names and asks you to enter one.

ctlfile
is the filename of the control file that is used to apply updates to the source file before text is generated. The filetype must be CNTRL. The IBM-supplied control files are described in Chapter 7, "How VM/XA System Product Uses Control Files and Update Files" on page 361.

options:

PPF
indicates that the third operand in the command is the name of a product parameter file or override file that specifies the minidisk/directory search order and the name of the control file.

CTL
indicates that the third operand in the command is the name of a control file. If CTL is specified, the product parameter file is not used.

If neither CTL nor PPF is specified, VMFNLS searches for a file whose name matches the third operand in the command. If a file with that name is located, VMFNLS checks the filetype to determine if the file is product parameter file ($PPF) or a control file (CNTRL). If both files exist, the product parameter file is used.

SETUP
sets up a minidisk/directory access order for the assemble function. This option is valid only when using a product parameter file.

NOSETUP
does not set up a new access order.

KEEPSRC
indicates that the merged source file consisting of the updates and the base source file will be saved on the user's disk. The file is named *$filename filetype*, where *filename* is the original source input filename and *filetype* is the original source input filetype.

NOKEEPSRC
does not save the merged source file.

Note: In addition to the options described above, VMFNLS accepts certain options for three of the commands that it issues:

- ASSEMBLE
- GENMSG
- CONVERT COMMANDS.

Some options are assigned by VMFNLS; other options that can be used depend on the filetype of the source file. See the following section, "How VMFNLS Works." For complete descriptions of these commands and their options, see *VM/XA SP CMS Command Reference*.

How VMFNLS Works

1. If a product parameter file is being used:
   a. VMFNLS calls the VMFOVER EXEC to build a temporary product parameter file (filetype $PPFTEMP) containing the parameters for the component being processed.
   b. If a user EXEC is specified in the USEREXIT tag in the $PPFTEMP file, VMFNLS calls the user EXEC with the parameters VMFNLS and SET-UP.
   c. If the NOSETUP option is not specified, VMFNLS calls the VMFSETUP EXEC with the ACCESS ASM options to save the current minidisk/directory access order and build a new one for the assemble function using data from the $PPFTEMP file.
2. VMFNLS issues the UPDATE command with the CTL, STK, PRINT, and OUTMODE A1 options.

   UPDATE uses the control file (*ctlfile* CNTRL) specified in the product parameter file or in the VMFNLS command to update the assembler language source file.

   UPDATE stacks information from the control file in the console stack and prints the update log file.

   The OUTMODE A1 option specifies that the files created by the UPDATE command are written on filemode A with a filemode number of 1 (A1).

   If a PTF file is missing, UPDATE writes a message to the console and to the update log file.
3. At this point, processing depends on the filetype of the input source file:
   - If the source file is an ASSEMBLE file:
     a. Using the library list from the MACS records in the control file, VMFNLS issues a GLOBAL MACLIB command.
     b. VMFNLS issues the ASSEMBLE command to assemble the updated source file.

- If the source file is a REPOS (message repository) file:
  a. VMFNLS determines the *langid* associated with the source file:
     - If the source filename is only 6 characters, VMFNLS assigns AMENG (American English) as the default *langid.*
     - If the source filename is longer than 6 characters, VMFNLS extracts the country code from the seventh and eighth characters of the filename and searches the VMFNLS LANGLIST file to find the corresponding *langid.* This file contains a list of all valid country codes, along with the associated *langids* and language names, as shown in Figure 52.

```
*   AMENG  TAM  American English
A   KANJI  TKA  Kanji
B   UCENG  TUC  Uppercase English
C   PORTG  TPO  Brazilian Portuguese
D   FRANC  TFR  French
E   GER    TGR  German
```

Figure 52. VMFNLS LANGLIST File

If the country code from the source filename is not contained in the VMFNLS LANGLIST file, all temporary files are erased, the printer is closed, and VMFNLS terminates.

b. VMFNLS issues the GENMSG command to compile the input REPOS file and produce a text file. VMFNLS sets up the operands for GENMSG as follows:

fn
is the filename of the file to be compiled. GENMSG places this name in the ESD card when the text file is produced. The possible filenames are:

fn	**Repository**
HCPMES	CP
DMSMES	CMS
CSIMES	GCS

ft
is REPOS, the filetype of the repository file to be compiled.

applid
corresponds to the first three characters of the filename of the source file:

applid	**Application**
HCP	CP
DMS	CMS
CSI	GCS

langid
is the language identifier of the source file.

options
depend on the component associated with the source file. For the CP repository, VMFNLS uses the GENMSG options `CP MARGIN 63`. For other components, VMFNLS uses the GENMSG option `MARGIN 63`.

Note: For a Kanji CP repository, VMFNLS uses the GENMSG options `CP DBCS`. For other Kanji repositories, VMFNLS uses the GENMSG option `DBCS`.

Other GENMSG options can be specified with the VMFNLS command if they do not conflict with the GENMSG options assigned by VMFNLS.

GENMSG produces a text file that has the same filename as the input file and a filetype of TEXT. GENMSG also produces a LISTING file, whose filename is the source file's filename with a prefixed dollar sign.

The following table shows some examples of the text and listing files written to filemode A from a given message repository source file:

Source File	Text File	Listing File
HCPMES REPOS	HCPMES {TEXT|TXT*nnnnn*}	$HCPMES LISTING
DMSMESA REPOS	DMSMESA {TEXT|TXT*nnnnn*}	$DMSMESA LISTING
CSIMESB REPOS	CSIMESB {TEXT|TXT*nnnnn*}	$CSIMESB LISTING

Note: *nnnnn* is the number of the most recently applied PTF.

The updated source file is then printed.

For a complete description of the GENMSG command, see *VM/XA SP CMS Command Reference*.

- If the source file is a DLCS (definition language for command syntax) file, VMFNLS issues the CONVERT COMMANDS command to produce two text files from the input file.

  VMFNLS sets up the operands for CONVERT COMMANDS as follows:

  fn
  is the filename of the file to be compiled.

  ft
  is the filetype of the file to be compiled.

  fm
  is always set to *.

  options
  are not assigned by VMFNLS; any valid CONVERT COMMANDS options can be specified.

  The filenames of the text decks produced by CONVERT COMMANDS depend on the DLCS statement contained in the input file. This statement identifies the *applid*, *langid*, and whether the input file is a user or system DLCS file. CONVERT COMMANDS uses the *langid* to obtain the corresponding country code (*y*) from the VMFNLS LANGLIST file.

  For a system DLCS file, the filenames of the text decks are *applid* SPA*y* for the command syntax definition file and *applid* SSY*y* for the translation and synonym table. For a user DLCS file, the filenames of the text decks are *applid* UPA*y* for the command syntax definition file and *applid* USY*y* for the translation and synonym table.

  The filetype of the output text files is TEXT.

The following table shows some examples of the text files produced from a given source file:

Source File	Text Files
DMSSPA DLCS	DMSSPA TEXT DMSSSY TEXT
DMSSPAB DLCS	DMSSPAB TEXT DMSSSYB TEXT
DMSSPAC DLSC	DMSSPAC TEXT DMSSSYC TEXT

The updated source file is then printed.

For a complete description of the CONVERT COMMANDS command, see the *VM/XA SP CMS Command Reference*.

4. Each output TEXT file from the ASSEMBLE, GENMSG, or CONVERT COMMANDS processing is concatenated with the UPDATES file so that the TEXT file contains a history of update activity.
5. VMFNLS renames the TEXT file as follows:
   - If the PPF option is specified and no PTF prefix is found in the control file, the TEXT file is renamed *fn* TXT*nnnnn*, where *nnnnn* is the PTF number obtained from last AUX file entry processed by the UPDATE command. If a PTF prefix is found in the control file, that prefix overrides the TXT default.
   - If the CTL option is used, VMFNLS uses the update level identifier from the control file (the identifier of the most recent update that was found and applied is stacked by the UPDATE command), to rename the TEXT file. If the update level identifier is anything other than TEXT, the TEXT file is renamed *fn* TXT*nnnnn* (where *nnnnn* is the 1- to 5-character update level identifier).

   The new *fn* TXT*nnnnn* or *fn* TEXT file is found on filemode A.

   Note: The new text deck will be filemode A1 regardless of the filemode of the original text deck. If the filemode number of the original text deck is anything other than 1, you must rename the text deck created by VMFNLS.
6. If a product parameter file is being used:
   a. VMFNLS calls VMFSETUP to restore the original minidisk/directory access order.
   b. If a user EXEC is specified in the USEREXIT tag in the $PPFTEMP file, VMFNLS calls the user EXEC with the parameters VMFNLS, CLEAN-UP, and a return code.

Messages

DMSWNL001E **No filename specified.**
DMSWNL002E **VMFNLS LANGLIST * not found.**
DMSWNL023E **No filetype specified.**
DMSWNL032E **Invalid filetype** *ft*.
DMSWNL122E **Return code** *rc* **from** *routine*.
DMSWNL179E **Missing or invalid MACS card in control file** *fn ft fm*.
DMSWNL328E **Control file not specified.**
DMSWNL448E **Country code** *code* **not in** *list*.

For a complete explanation of each message, see *VM/XA SP System Messages and Codes Reference*.

VMFOVER EXEC

The VMFOVER EXEC makes a temporary copy of one component section from the product parameter file with changes made in accordance with the override section of the product parameter file or a separate PPF override file. The other service and installation EXECs use this copy instead of the base product parameter file.

VMFREC, VMFAPPLY, VMFHASM, and VMFBLD all issue VMFOVER automatically. You can also issue VMFOVER from the command line if you want to see the temporary product parameter file.

Format

The format of the VMFOVER EXEC is:

VMFOVER	*prodid compname*

Where:

prodid
is the product identifier of the product to be serviced. For VM/XA System Product Release 2.1, the product identifier is 56643089.

compname
is the component to be serviced, for example, CP or CMS. *compname* can be a name listed after the :COMPLST. tag or the :OVERLST. tag in the base PPF; or after the :OVERLST. tag in the PPF override file. If you do not specify the *compname*, you will be shown a list of components and asked to choose one.

How VMFOVER Works

VMFOVER locates the file with a filename of *prodid* and a filetype of $PPF. This file can be either a base PPF or a PPF override file. The EXEC then checks the list of component names following the :COMPLST. tag for the component name you specified. If the name is found, this is the base product parameter file. The appropriate component section is copied to a file called *prodid* $PPFTEMP, and processing stops.

If the component name is not listed after the :COMPLST. tag, VMFOVER looks for it after the :OVERLST. tag. If the component name is not listed there, VMFOVER issues an error message.

If the component name is found after the :OVERLST. tag, VMFOVER looks for a *:compname.* tag. The *:compname.* tag tells VMFOVER where to look for the basic information about the component. For example:

```
:CP. CP 56643089

:CORCP. CP
```

The first example is from an override file. It tells VMFOVER that the basic information about CP is found in the CP section of 56643089 $PPF. The second example is from the override section of a base PPF. Notice that it does not specify a filename. It tells VMFOVER to look for the base information about CORCP in the CP section of *this file.*

Override information follows the *:compname.* tag. VMFOVER combines the override information with the base information to create a copy of the appropriate component section of the product parameter file. The filename of the copy is the filename of the override file if the NEWNAME option was specified on the *:compname.* tag in the override file; otherwise, it is the filename of the base PPF. The filetype of the copy is $PPFTEMP.

The NEWNAME option on the :compname. tag is reserved for creating a test or sample override file executing VMFOVER from the command line. For any other execution, the filename of the temporary product parameter file must match the prodid or product identifier of the product on the tape to be serviced and NEWNAME should not be specified.

For examples of a base product parameter file with an override section, an override file, and $PPFTEMP files, see Chapter 8, "Files Used in PUT and COR Service."

VMFPLC EXEC

VMFPLC is a common EXEC which calls either VMFPLC2 or VMFPLCD. VMFPLC supports the VMFPLC2 and VMFPLCD command formats. A ROUTE function allows VMFPLC to establish command routing for subsequent VMFPLC2 or VMFPLCD commands issued by VMFPLC.

VMFPLC acts as a front-end processor to minimize changes to existing routines which use VMFPLC2 when conversion to VMFPLCD or a dual path is desired. It allows either command format to be used including a mixture of the two sets of functions, and parameters and options.

Format

The format of the VMFPLC EXEC is:

VMFPLC	ROUTE *options:* *command string for VMFPLC2 or VMFPLCD* *options:* DISK [*fn ft fm*] TAPE

Where:

ROUTE
is *not* passed to VMFPLC2 OR VMFPLCD. It is used to establish the routing for subsequent commands issued to VMFPLC which are to be passed to VMFPLC2 or VMFPLCD, or to reset any previous routing established.

If ROUTE is not specified, it is assumed that you are entering a VMFPLC2 or VMFPLCD command string.

If no option (DISK or TAPE) is specified with the ROUTE operand, ROUTE resets the routing variable to TAPE, regardless of any previous setting.

options:

DISK [*fn ft fm*]
establishes routing to disk (VMFPLCD). If the routing is to disk, the file ID of the envelope may also be established (optional). If the envelope file ID is provided, it must be the complete file ID of *fn ft fm*. The file ID is checked for validity only to the extent that there are three parameters.

TAPE
establishes routing to tape (VMFPLC2).

Usage Notes

1. When you issue VMFPLC, and not VMFPLC2 or VMFPLCD directly, VMFPLC will route to VMFPLC2 or VMFPLCD according to the routing in the VMFPLC GLOBALV variable from a VMFPLC ROUTE command. If there is no routing established, the default routing is VMFPLC2.

2. Once the command routing has been determined, the options and control functions will be converted (if necessary) to those appropriate for the chosen routing as follows:

   a. Options WTM, NOWTM, EOT and EOF from VMFPLC2, and WGS, NOWGS, EOD and EOG from VMFPLCD will be converted to the respective keyword for the command to be called.

      VMFPLC2 options TAP*n*, CUU (device address), TRTCH *a* DEN *den* and *nn*TRACK will be deleted if the routing is to VMFPLCD.

   b. The control functions will be converted as follows:

VMFPLC2	VMFPLCD	
MODESET	No equivalent (See Note 1)	
ERG	No equivalent (See Note 1)	
B/FSR	No equivalent (See Note 2)	
B/FSF	B/FSG	
REW	RST	
RUN	RST	(See Note 3)
WTM	WGS	

      Notes:

      1) VMFPLC2 MODESET and ERG functions have no equivalent function in VMFPLCD and will be handled as a "no operation" request by VMFPLC if the command is for disk. For example, a return code of zero will be passed back without calling VMFPLCD.

      2) VMFPLC2 BSR and FSR have no equivalent function in VMFPLCD. The functions will be passed unchanged to VMFPLCD (if the medium is DISK) resulting in an error because these functions are supposed to alter the position within the tape or envelope and no position change occurs.

      3) VMFPLC2 REW and RUN functions both convert to a VMFPLCD RST function. The VMFPLCD RST function converts to the VMFPLC2 REW function, **not** the RUN function.

3. VMFPLC will not check command syntax or option validity. With the exception of the previous keyword/command handling, the command will be passed intact. Thus, all errors are issued by VMFPLC2 or VMFPLCD.

4. If the envelope file ID is provided with the DISK option, it is used to set or reset the GLOBALV variable used by VMFPLCD to remember the file ID across multiple invocations. If the file ID is different from that in an existing GLOBALV, or there is no existing GLOBALV, the logical record position in the envelope is set to record 1. If the GLOBALV variable exists and the file ID on the ROUTE function is the same as in the GLOBALV, no change in the logical record pointer is made.

5. Routing information is maintained in a GLOBALV variable for VMFPLC across multiple invocations of VMFPLC. The GLOBALV variable is maintained only within a CMS IPL; it is reset automatically on IPL. If ROUTE is issued with no parameters, the VMFPLC GLOBALV routing variable is deleted. This does not reset the file ID for the envelope file used by VMFPLCD.

Messages and Return Codes

DMSWPC0003E Invalid option *option* [RC=24]
DMSWPC1447E Invalid command format [RC=24]

.

Note: Any other messages you may receive are returned from the VMFPLC2 EXEC or VMFPLCD EXEC. See the VMFPLC2 EXEC or VMFPLCD EXEC descriptions for an explanation of these messages.

VMFPLCD EXEC

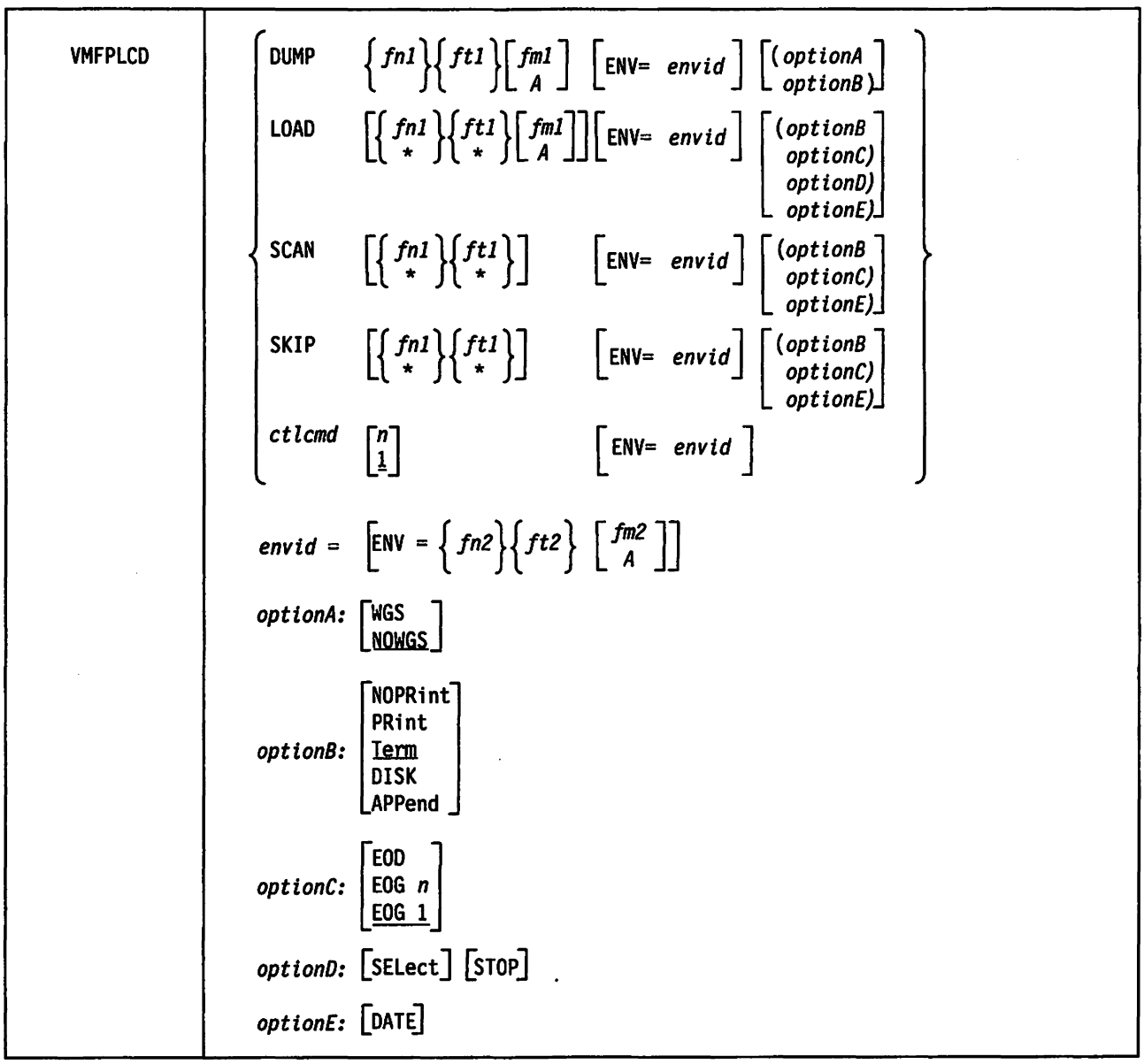
Use the VMFPLCD EXEC to:

- Load files from the envelope file
- Dump CMS files to the envelope file (files to be dumped can contain either fixed-length or variable-length records)
- Load previously dumped files from the envelope file
- Do various control operations on the envelope file.

Note: Files processed by the VMFPLCD EXEC must be CMS-formatted.

Format

The format of the VMFPLCD EXEC is:



Where:

DUMP
dumps one or more CMS files to an envelope file. You must specify a *fn1* and *ft1*; *fm1* is optional. They may be specified using "wild card" characters, as described in the "Usage Notes" number 12.

If you specify *fm1* as a letter, that filemode and its extensions are searched for the specified files. If you specify *fm1* as a letter and number, only files with that filemode number on that filemode and its extensions are dumped. If you use wild card substitution, specifying *fm1* as an asterisk (*), all filemodes are searched for the specified files. If you do not specify the *fm1*, only filemode A and its extensions are searched.

The envelope file that is to receive the dumped file is identified by the *ENV= fn2 ft2 fm2* parameters. See the "Usage Notes" for a description of the default assignment if the *ENV=* parameter is not supplied. The *fm2* must be for a disk accessed in write mode.

Note: The envelope file may not be dumped to itself. See "Usage Notes" for a more detailed description.

LOAD
copies the specified files contained in an envelope file created by DUMP into separate disk files with the file identifier of the original files. The envelope file ID is identified by the *ENV= fn2 ft2 fm2* parameter.

You may specify *fn1* and *ft1* with wild card characters as described in the "Usage Notes."

Note: No file may be loaded from the envelope which would overlay the envelope. See the "Usage Notes" for a complete description.

Option C (EOG or EOD) controls the limit of the search in the envelope file for a disk file to be loaded.

If a file identifier (*fn1 ft1*) for the file to be loaded is specified, only that one file is copied from the envelope file. If no file identifier is specified, all files within the *n* file groups from the option EOG n (or in the entire envelope file for option EOD) are copied. The *fn1* and *ft1* may be specified using wild card substitution characters. If wild card substitution is used, then any file in the envelope that satisfies the *fn1 ft1* specified will be loaded within the limits set by options EOG n or EOD. See the description for wild card specification in the "Usage Notes."

The files are written to the disk identified by the filemode letter. This disk must be accessed as read-write mode. If a filemode number is specified, only files with that filemode number are loaded. The default filemode is A.

SCAN
positions the envelope file at a specified point and lists the identifiers of the files within the envelope file scanned. Scanning occurs over *n* file groups, as specified by the option EOG *n* (the default is 1) or over the entire envelope file (EOD option).

If a specific disk file identifier (*fn1 ft1*) is specified, scanning stops upon encountering that file. The position within the envelope file is set to the header record of the file which satisfies the file identifier.

The *fn1* and *ft1* may be specified using wild card substitution characters. See the "Usage Notes" for details. If wild card substitution symbols are used, the file identifier is not used to locate a specific file on which to stop the scan for positioning. Instead, the file identifier is used only to determine which files will be logged.

If *fn1* and *ft1* are not provided, all files encountered are logged if logging is requested.

SKIP
positions the envelope file at a specified point and lists the identifiers of the files within the envelope file skipped. Skipping occurs over *n* file groups, as specified by the option EOG *n* (the default is 1 file group) or over the entire envelope file (EOD option).

If a specific disk file identifier (*fn1 ft1*) is specified, skipping stops upon encountering that file. The position within the envelope file is set to the header record of the next file, or the group separator record, whichever follows.

The *fn1* and *ft1* may be specified using wild card substitution characters. If wild card substitution symbols are used, the file identifier is not used to locate a specific file on which to stop the scan for positioning. Instead the file identifier is used only to determine which files will be logged. See the "Usage Notes" for a description of wild card substitution.

If *fn1* and *ft1* are not provided, all files encountered are logged if logging is requested.

ctlcmd
provides logical positioning and file group separation functions for the envelope file.

The functions are:

tapcmd	Action
BSG	Backspace *n* file group separators
FSG	Forward space *n* file group separators
WGS	Write *n* file group separators
RST	Reset logical position to start of the file envelope file, and reset the file ID of the envelope file.

ENV=
The ENV= parameter represents the envelope file that is to be processed. The envelope is represented as *fn2 ft2 fm2*. See the "Usage Notes" for a description of the default assignment if the ENV= parameter is not specified. *fm2* must be for a disk accessed in write mode.

***optionA*:**

WGS
writes a file group separator after each file is dumped.

NOWGS
indicates that no group separator is written after each file is dumped. NOWGS is the default.

***optionB*:**

NOPRint
does not list the files dumped, loaded, scanned or skipped.

PRint
spools the list of files dumped, loaded, scanned, or skipped to the printer.

Term
displays a list of files dumped, loaded, scanned, or skipped at the terminal. TERM is the default.

DISK
creates a CMS file called DISK MAP A5 containing the list of files dumped, loaded, scanned, or skipped.

APPend
causes the disk file containing the list of files dumped loaded, scanned, or skipped to be added to the end of an existing list file (DISK MAP A5).

***optionC*:**

EOD
reads the envelope file until the end-of-disk condition is recognized. End-of-disk is signaled by either two successive file group separators or an end-of-file condition during a read of the envelope file.

EOG *n*
EOG 1
reads the envelope file through a maximum of *n* file group separators (end-of-group records). The default is EOG 1.

optionD:

SELect
inhibits loading of a file from the envelope file that causes replacement of an identical file in virtual direct access storage. Files will be loaded only if they do not exist on the disk specified by the LOAD command, or when the date/time stamp for the file being loaded **does not match** the date/time stamp for the existing file.

STOP
assumes that files contained in the envelope file are in alphabetical sequence by filename and filetype. If the requested file is found, the file is loaded and the envelope file is positioned at the next record following the loaded file. If the file is not in the envelope and a file is encountered that is alphabetically beyond the bounds of the requested file, no file is loaded and the envelope file is positioned to load this file. You must specify the disk file *fn1* and *ft1* without using wild card substitution characters. If either the *fn1* or the *ft1* contains substitution characters, the STOP option will be ignored.

optionE:

DATE
displays LISTFILE information during a load, scan or skip function. The information displayed includes the record format, logical record size, number of records, and date/time stamp.

Usage Notes

1. All control functions for *ctlcmd*, except RST, allow a repetition factor (*n*) to be specified, to control the number of groups skipped or the number of group separator records to be written. The default is 1.
2. The file ID of the envelope file is controlled by two separate factors: a GLOBALV variable saved between VMFPLCD invocations, and the ENV= parameter.
   - If the ENV= parameter is provided, it specifies a file ID for the envelope file which takes precedence over the other methods of specification. In this case, the *fn2* and *ft2* must be provided (wild card substitution is not allowed). The *fm2* is optional, and has a default of A.
   - If the ENV= parameter is not specified, the GLOBALV variable is checked to find the file ID of the envelope file. If present, the GLOBALV variable for the file ID will be used. This allows the first invocation of the command to set the file ID and all subsequent commands for the same envelope file to be entered without the ENV= parameter.

     Note: All functions except RST, DUMP and WGS require the envelope file to exist. If it does not, the function will fail and the GLOBALV variable will not be updated with the envelope file ID.

   - If there is no ENV= parameter and no GLOBALV variable, an error will be returned.
3. The GLOBALV variable is maintained for one envelope file at a time, to remember the file ID and the logical position across multiple invocations of VMFPLCD. When a new envelope file ID is used, the variable is reset automatically to the logical start of the envelope file. The variable is maintained only within a CMS IPL. It is reset automatically on each CMS IPL to null.
4. If you wish to change the envelope file ID within the same IPL, the ENV= parameter should be used. Except for RST, DUMP or WGS functions, the envelope file must exist. Otherwise, the EXEC will end with an error, and the GLOBALV variable will not be updated.
5. If the prior envelope file is erased, and then a VMFPLCD command is issued with no ENV= parameter, the RST, DUMP or WGS commands will establish a new envelope with the same name as the prior one.
6. If this EXEC issues an error with a return code other than 24, the GLOBALV variable is updated. On input/output errors, the position is set to the last known position prior to the error.
7. Since the end-of-disk condition is recognized as either two successive group separator records or the physical end of the envelope file, logical positioning functions involving the end-of-disk condition will be handled as if there were two group separator records at the end of the envelope file. This provides a

consistent external result matching VMFPLC2, regardless of what is actually present. The actual end of the envelope file could have no, one or two real group separators. If there are less than two real group separators, the required number of positions beyond the last record will be treated as group separators to satisfy this requirement.

8. BSG will scan the envelope backwards to locate a group separator record and will leave the position ready to read this group separator. FSG will scan forward to locate a group separator and leave the position ready to read the following record.

9. If you wish to reset the logical position to the start of the envelope file, use the RST control function.

10. The LOAD, SKIP and SCAN functions leave the position after the file or the group separator that satisfied the conditions specified. SCAN can also leave the position ready to read a specific file.

11. DUMP and WGS leave the position ready to write at the next record position after the dumped files or group separator records written as a result of the command. If the logical position at the start of a DUMP or WGS separator is other than the next record past the actual physical end of the envelope, then the envelope will be adjusted prior to the function, to either truncate it or convert the missing group separator records to real records.

12. When specifying the *fn1 ft1* or *fm1* identifiers for the disk files, special wild card characters may be used, where indicated in the option and operand descriptions, to allow any characters to satisfy all or part of the identifier. The character * represents any number of characters including zero. As many asterisks as required can appear anywhere in a filename or filetype. Only one asterisk can be used for filemode. The character % is a place-holding character that means a single character, but any character will do. As many percent symbols as necessary may appear anywhere in a filename or filetype. The percent symbol may not be used for filemode.

    When wild card substitution is used during a DUMP function to select the files to dump, the order in which the files are dumped is the same order in which the files would be found with a CMS LISTFILE command. This order can vary from session to session. Thus, if a specific order is needed, do not use wild card substitution.

13. The *fn ft fm* parameters (disk file ID or envelope file ID) must conform to CMS rules. For example, the filename and filetype must be eight positions and the filemode is two positions. If they are longer, they will be truncated to the maximum length and no error message will be issued.

14. Conflicting options (for example, WGS and NOWGS) will result in an error from VMFPLCD. This is not treated as an error with VMFPLC2. Instead the last value entered is used.

15. VMFPLCD dumps files that have a *fn1* or *ft1* containing mixed-case characters only if wildcard symbols are used in the file ID. For example, the file Test FILE can be dumped with a file ID of T%%% FILE, * FILE, T* *, or other such IDs, but will fail if requested as Test FILE.

16. If an attempt is made to dump the envelope file (or a copy of it) to itself, or if a file to be loaded from the envelope would overlay the envelope, processing is stopped and an error occurs.

17. An envelope may contain other envelope files, but not itself. All control records within the envelope contain a prefix section which identifies the envelope. This prefix is established on the first output function performed and remains the same even if the envelope file is renamed in the middle of processing. Two different envelope files that were created with the same *fn* and *ft* will appear to VMFPLCD as the same envelope even if one is renamed after creation.

18. The envelope file created by VMFPLCD is not packed to condense the size. Packing of the envelope prior to sending it to another user will greatly reduce the size of the file. Before processing the envelope again, it must be unpacked.

19. The keyword ENV= must always be followed by a blank.

Messages and Return Codes

DMSWPD0002E	File *VMFPLCM EXEC* * not found [RC=28]
DMSWPD0002E	File *fn ft fm* not found [RC=20|28|36]
DMSWPD0003E	Invalid option: *option* [RC=24]
DMSWPD0010S	Premature end occurred on *fn ft fm* [RC=40]
DMSWPD0014E	Invalid function *function* [RC=24]
DMSWLM0014E	Invalid function *function* [RC=24]
DMSWPD0029E	Invalid parameter *parameter* in the option *option* field [RC=24]
DMSWPD0040E	No files loaded [RC=28]
DMSWPD0048E	Invalid filemode *filemode* [RC=24]
DMSWPD0066E	*option option option option option* are conflicting options [RC=24]
DMSWPD0069E	Filemode *filemode* not accessed [RC=36]
DMSWPD0104S	Error *rc* reading file *fn ft fm* from disk or directory [RC=100]
DMSWPD0105S	Error *rc* writing file *fn ft fm* on disk or directory [RC=100]
DMSWPD0671E	Error loading file *fn ft fm*; rc=*rc* from COPYFILE [RC=100]
DMSWLM0814E	Message number *num*, format *fmt* was not found [RC=12]
DMSWLM0814E	Dictionary number *num* format *fmt* not found [RC=12]
DMSWPD1447E	Invalid command format [RC=24]
DMSWPD1448E	Option(s) *option option option option option option option option option option option option* invalid for *function* [RC=24]
DMSWPD1450S	Output file *fn ft fm* disk is read-only [RC=36]
DMSWPD1451S	Cannot dump an envelope file to itself. File *fn ft fm* is same envelope file [RC=32]
DMSWPD1452S	Unidentifiable envelope control record in *fn ft fm* [RC=100]
DMSWPD1453S	Loading of file *fn ft fm* would overlay envelope [RC=32]
DMSWPD1454S	Date/time stamp update on *fn ft fm* failed [RC=104]
DMSWPD1455S	Error occurred trying to truncate file *fn ft fm* [RC=100]
DMSWPD1456S	Unexpected error *rexx code* from VMFPLCD [RC=104]
DMSWPD1457E	Envelope file was not specified [RC=24]
DMSWPD1458E	File designated as envelope on *function* is not an envelope [RC=32]
DMSWPD1459S	Number of records for file *fn ft fm* different than when dumped [RC=40]
DMSWPD1460E	Envelope file *fn ft* exists but is on R/O extension of *fm* disk [RC=36]

For a complete explanation of each message, see *VM/XA SP System Messages and Codes Reference.*

VMFPLC2 Command

The VMFPLC2 command loads source code files from the Product Tape and loads the service installation EXEC (VMSERV) from the PUT. VMFPLC2 also dumps CMS files from disk to tape, loads previously dumped files from tape to disk, and performs various control operations on a specified tape drive. Disk files to be dumped can contain either fixed-length or variable-length records.

The VMFPLC2 command does not process multivolume files. Files processed by the VMFPLC2 command must be in a unique CMS format.

Format

The format of the VMFPLC2 command is:

```
VMFPLC2  { DUMP    {fn}|* {ft}|* [fm]|* [(optionA optionB optionD)]
         { LOAD    [{fn}|* {ft}|* [fm]|A] [(optionB optionC optionD optionE)]
         { SCAN    [{fn}|* {ft}|*] [(optionB optionC optionD optionF)]
         { SKIP    [{fn}|* {ft}|*] [(optionB optionC optionD)]
         { MODESET [(optionD)]
         { tapcmd  [n|1] [(optionD)]

optionA: [WTM|NOWTM]

optionB: [NOPRint|PRint|Term|DISK|APPend]

optionC: [EOT|EOF n|EOF 1]

optionD: [TAPn|cuu|181] [DEN den]

optionE: [SELect] [STOP]

optionF: [DATE]
```

Where:

DUMP {*fn*} {*ft*} {*fm*}
{* } {* } {* }

dumps one or more disk files to tape. If *fn* or *ft* is specified as an asterisk (*), all files that satisfy the other file identifier are dumped.

If *fm* is coded as a letter, that disk and its extensions are searched for the specified files. If *fm* is coded as a letter and number, only files with that mode number and letter (and the extensions of the disk referenced by that *fm* letter) are dumped. If *fm* is coded as an asterisk (*), all accessed disks are searched for the specified files. If *fm* is not specified, only the A-disk and its extensions are searched.

LOAD {*fn*} {*ft*} {*fm*}
{* } {* } {A }

reads CMS files onto disk. The *optionC* options control the area on the tape that the command searches. For instance, if EOF 1 (the default) is in effect, VMFPLC2 loads only the files specified before the first physical tape mark.

If a file identifier is specified, only that one file is loaded. If an asterisk (*) is specified for *fn* or *ft*, VMFPLC2 loads all files that satisfy the other file identifier.

The files are written to the disk indicated by the filemode letter. The filemode number, if entered, indicates that only files that have that filemode number are to be loaded.

SCAN {*fn*} {*ft*}
{* } {* }

lists the identifiers of the files it scans. The *optionC* options control the area on the tape that the command searches. For instance, if EOF 1 (the default) is in effect, VMFPLC2 scans only the files specified before the first physical tape mark. However, if a file identifier (*fn* and *ft*) is specified, scanning stops upon encountering that file; the tape remains positioned ahead of the file.

SKIP {*fn*} {*ft*}
{* } {* }

lists the identifiers of the files it skips. The *optionC* options control the area on the tape that the command searches. For instance, if EOF 1 (the default) is in effect, VMFPLC2 skips only the files specified before the first physical tape mark. However, if a file identifier (*fn* and *ft*) is specified, skipping stops after encountering that file; the tape remains positioned immediately following the file.

MODESET

sets the values specified by the DEN, TRACK, and TRTCH options. After initial specification in a VMFPLC2 command, these values remain in effect for the virtual tape device until they are changed in a subsequent VMFPLC2 command, RDTAPE, WRTAPE, or TAPECTL macro.

tapcmd [*n* / 1]

specifies a tape control function (*tapcmd*) to be executed *n* times (default is 1 if *n* is not specified). These functions also work on tapes in a non-CMS format.

tapcmd	Action
BSF	Backspace *n* tape marks
BSR	Backspace *n* tape records
ERG	Erase gap
FSF	Forward-space *n* tape marks
FSR	Forward-space *n* tape records
REW	Rewind tape to load point
RUN	Rewind tape and unload
WTM	Write *n* tape marks.

***optionA*:**

WTM
writes a tape mark on the tape after each file is dumped.

NOWTM
writes a tape mark after each file is dumped, then backspaces over the tape mark so that subsequent files written on the tape are not separated by tape marks.

***optionB*:**

NOPRint
does not spool to the printer the list of files dumped, loaded, scanned, or skipped.

PRint
spools to the printer the list of files dumped, loaded, scanned, or skipped.

Term
displays a list of files dumped, loaded, scanned, or skipped.

DISK
creates a disk file containing the list of files dumped, loaded, scanned, or skipped. The disk file has the file identification of TAPE MAP A5.

APPend
causes the disk file (containing the list of files dumped, loaded, scanned, or skipped), to be added to the end of an existing TAPE MAP.

***optionC*:**

[EOT]
[EOF *n*]
[EOF 1]
EOT reads the tape until an end-of-tape indication. EOF *n* reads the tape through a maximum of *n* tape marks. EOF 1 is the default.

***optionD*:**

[TAP*n*]
cuu
181]

specifies the virtual device address of the tape to be read or written to; *cuu* is 181, 182, 183, or 184. The default is 181. You can also use TAP*n*, where *n* is a single digit number. The number is appended to the number 18 to form a device number. For instance, TAP1 becomes 181 and TAP2 becomes 182. The unit specified by *cuu* must previously have been attached to your CMS virtual machine before any tape I/O operation can be attempted.

DEN *den*
is the tape density, where *den* is 800, 1600, 6250, or 38K. 9TRACK is assumed. In the case of either 800/1600 or 1600/6250 dual-density drives, 1600 is the default.

***optionE*:**

SELect
inhibits loading of a file from the tape in cases when VMFPLC2 would replace an *identical* file on the disk. Files will be loaded only if they do not exist on the specified disk, or when the date/time stamp for the file on the disk does *not match* the date/time stamp for the corresponding file on the tape.

STOP
assumes that files contained on the tape are in alphabetical sequence. If the requested file is on the tape, the file is loaded onto disk and the tape stops. If the file is not on the tape and a file is encountered that is alphabetically beyond the bounds of the requested file, the tape stops. You must specify *fn ft*. Neither *fn* nor *ft* may be specified as an asterisk (*).

***optionF*:**

DATE
displays listfile information during a SCAN. The information displayed includes number of records, length of records, and date/time stamp.

Usage Notes

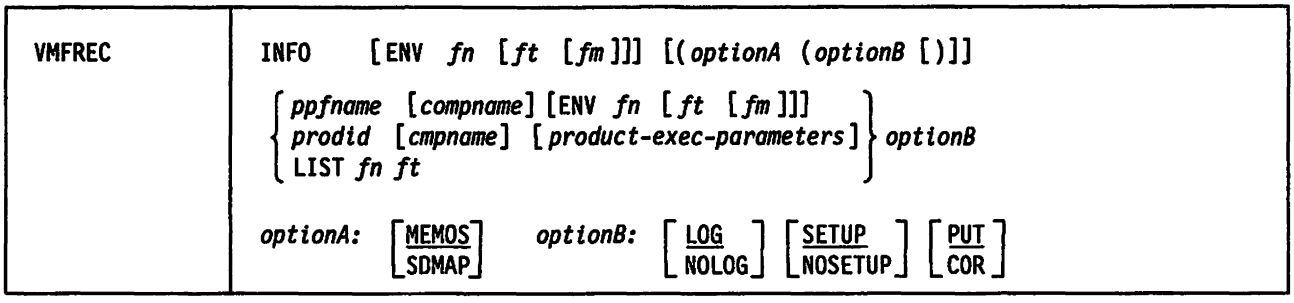
1. If conflicting options are specified on the VMFPLC2 command line, the last option entered on the command line is in effect.
2. The VMFPLC2 command writes tape records 4005 bytes long. The first character is a binary 2 (X'02'), followed by the characters CMS and a file format byte, followed by 4000 bytes of file data packed without regard for logical record length. If a null block is dumped, the character 0 replaces the byte after CMS. This causes subsequent loading of null blocks to be ignored. In the final record, the character N replaces the blank after CMS, and the data area contains CMS file directory information.
3. If a tape file contains more CMS files than would fit on a disk, the tape load operation may terminate if there is not enough disk space to hold the files. To prevent this, when you dump the files, separate logical files by tape marks, then forward space to the appropriate file.
4. The CMS file directory is the first record of each CMS file on tape.
5. It is possible to run a tape off the reel in at least one situation. If you specify EOF *n* and *n* is greater than the number of tape marks on the tape, the tape will run off the reel.
6. For more information on tape file handling, see *VM/XA SP Real System Operation*.
7. Do not use TAPE as a synonym of VMFPLC2, or vice versa. If you do, do not use these synonyms to call either function from within another EXEC. You may use any other valid synonym (for example, TAP).

VMFREC EXEC

Use the VMFREC EXEC to map a service tape and receive service files.

Format

The format of the VMFREC EXEC is:



Where:

INFO
processes the first two tape files using the default operand MEMOS. This operand creates a service diskmap for the volume mounted, if it does not already exist.

ENV *fn* [*ft*[*fm*]]
if specified, indicates that the PTF medium will be an envelope. A filename must be specified with it. If a filetype is not specified, the default filetype SERVLINK will be used. If a filemode is not specified, then the first occurrence of the filename and filetype specified will be used.

If a filemode is specified, a LISTFILE is executed for the envelope. If the envelope is found, then VMFPLC ROUTE is executed so that all subsequent VMFPLC commands will be routed to the envelope. This allows the syntax of all VMFPLC commands in VMFREC and its part-specific EXECs to remain unchanged, except to call VMFPLC instead of VMFPLC2.

If a filemode is not specified, then LISTFILE is executed. If no occurrence of the envelope is found, an error message is issued. If LISTFILE locates an envelope, then the filemode of this envelope is saved and VMFPLC ROUTE is executed for that envelope.

After a valid envelope has been located and VMFPLC has been routed, a variable is set that contains the disk address of the envelope.

If ENV is not specifed, the TAPE option of VMFPLC ROUTE is executed.

After VMFREC calls VMFSETUP to perform accesses according to the PPF, a check is made to verify that the envelope is still located at the same filemode indicated by the previous VMFPLC ROUTE command. If the mode of the envelope disk was changed by VMFSETUP, VMFPLC ROUTE is executed, indicating the new filemode of the envelope. When VMFREC calls VMFSETUP to restore the disk setup, the check for the envelope location is again made and another VMFPLC ROUTE is issued if required.

Note: The PTF envelope must be on a disk that is accessed at the start of VMFREC. This disk must remain accessed during VMFREC processing. This means that the address of the disk containing the PTF envelope must be the A-disk or be listed on the SYSTEM string for the component being received in the product parameter file. If the filemode of the disk containing the PTF envelope is changed due to internal VMFREC disk access processing, a new VMFPLC ROUTE command will be issued so that the PTF envelope location remains known to VMFREC. In the same manner, if after VMFREC processing the restoration of the original disk access order results in a

change of filemode for the PTF envelope disk, another VMFPLC ROUTE command will be issued for the envelope at its new filemode.

ppfname
is the filename of the base product parameter file (which is the product identification number, *prodid*) or the filename of a product parameter override file. The filetype must be $PPF.

compname
is the name of the component.

For a product that uses a product parameter file, this could be the name of a base component (such as CP or CMS) or the name of a component parameter override area containing alternative or additional parameters for the component. If you do not specify a valid component or override name, VMFREC shows you a list of names and asks you to choose one. You can specify only one name.

prodid
is the product identification number of a product that does not use a product parameter file. It is also the filename of the product-specific exec. The filetype must be EXEC.

product-exec-parameters
are parameters that you want to pass to the product-specific exec.

LIST ***fn ft***
specifies the filename and filetype of a list of products or components (or both) that you want to service. Each entry in the list must use one of the following formats:

ppfname [*compname*]

prodid [*compname*] [*product-exec-parameters*]

Any options that you want to use must be entered on the VMFREC command, not inside the list itself.

Note: Options must apply to all products or components (or both) in the list.

***optionA*:**

MEMOS
transfers the tape descriptor file, the tape document, memos, and **crllnnf** files to your C-disk if it is accessed, otherwise it transfers them to your A-disk. MEMOS is the default option for INFO.

SDMAP
transfers the tape descriptor file.

***optionB*:**

LOG
writes VMFREC messages into a receive exception log ($VMFREC $ERRLOG).

Note: Messages issued by the VMFOVER EXEC (called by VMFREC) are not written into the log but are displayed at the terminal only.

NOLOG
displays VMFREC messages at the terminal only.

SETUP
sets up a minidisk/directory access order for the receive function according to entries in the MDA section of the product parameter file.

NOSETUP
does not set up a new access order.

Note: The SETUP/NOSETUP options are ignored with the INFO operand or to process products that do not use a product parameter file.

PUT
loads the files from a program update tape, if it is mounted.

COR
loads the files from a corrective service tape, if it is mounted.

How VMFREC Works

The following list summarizes VMFREC processing:

- INFO is used to selectively load tape files 1 and 2. When called with the INFO operand with the default option MEMOS, VMFREC does the following:
  1. VMFREC creates the tape map on filemode C (if it exists), or on filemode A.
  2. If a new service diskmap needs to be created, VMFREC loads the *crllnnf* files.
  3. VMFREC creates or appends to the service diskmap for this volume.
  4. Loads the documentation CMS files from tape files 1 and 2 to filemode C (if it exists) or filemode A.
- When called with the SDMAP option:
  - Since the purpose of this option is to create the service disk map, the steps for this option are already done at the beginning of VMFREC processing.
- When called with a *ppfname* or a *prodid*, either directly or through the LIST option, if the service tape has not been mapped, VMFREC maps the tape and loads the service documentation. Then VMFREC does the following processing for the specified *ppfname* or *prodid* (or, if the LIST option is used, for each product or component in the list):
  1. VMFREC determines if the product or component uses a product parameter file by looking at the filetype of the level identifier (located in the second tape file):
     - If the filetype is *crllnn*1, the product or component uses a product parameter file. The rest of this section describes the processing for products and components that use a product parameter file.
     - If the filetype is anything else, the product or component uses a product-specific EXEC. VMFREC does steps 5 and 6, then calls the product-specific EXEC and passes any parameters entered with the command. For more information about product-specific EXEC processing, see the product documentation.
  2. VMFREC calls the VMFOVER EXEC to build a temporary product parameter file (filetype $PPFTEMP) containing the parameters for the component being processed.
  3. If a user EXEC is specified in the USEREXIT tag in the $PPFTEMP file, VMFREC calls the user EXEC with the SETUP parameter.
  4. If the NOSETUP option is not specified, VMFREC calls the VMFSETUP EXEC with the ACCESS REC options to save the current minidisk or directory access order and build a new one for the receive function using data from the $PPFTEMP file. The access order is:

     TASK
     APPLY
     DELTA
     BASE
     BUILD
     SYSTEM.
  5. VMFREC checks the PUT or COR option specified with the command (PUT is the default) against the tape descriptor file (located in the first tape file) to make sure that the correct tape is mounted for the type of receive you want to do.
  6. VMFREC positions the tape to the files for the specified product or component.

7. VMFREC does merge processing as follows:
   a. VMFREC locates the MERGE tag in the $PPFTEMP file. The MERGE tag might list one or more names of minidisk/directory strings, such as DELTA1 or LOCAL1.
   b. VMFREC then locates the record for the first string name in the MDA section of the $PPFTEMP file. This record lists the minidisks or directories in the string in search order from highest to lowest, left to right.
   c. Beginning with the lowest search order pair in the string (the last two minidisks or directories listed on the right), VMFREC:
      1) Calls the COPYFILE command with the REPLACE option to copy all the files from the higher of the pair in the search order to the lower.
      2) Erases the higher of the pair.
      3) Selects a new pair consisting of the higher of the previous pair and the next higher in the string.
      4) Repeats the above steps for the new pair.
      5) Continues selecting pairs until the highest minidisk or directory in the string has been merged. This results in an empty minidisk or directory at the highest level.
   d. VMFREC looks at the next string name (if any) in the MERGE record and repeats the merge processing for that string.
8. VMFREC calls one or more of the following part-specific EXECs to process the service tape and load the service files. The next tape file on the service tape contains a product contents directory that lists the service files. The RECSER section of the $PPFTEMP file identifies the part-specific EXEC that handles each type of file and specifies the target string. The MDA section of the $PPFTEMP file identifies the specific target minidisk or directory in the string.

   VMFRCAXL This EXEC receives the apply lists and exclude lists.

   VMFRCUPD This EXEC receives source update files as follows:
   a. VMFRCUPD does not load any update files from the tape if they exist on the target disk and the update is not an update shell.
   b. VMFRCUPD searches the other accessed minidisks and directories for files with the same filename and filetype as the update files that have just been received.
      - If it finds an update shell with the same name, VMFRCUPD erases the update shell, but retains the dependent information and the filemode number.
      - If it finds a complete update file with the same name, VMFRCUPD erases the new file.

   VMFRCUPP This EXEC does the following with uppercase English HELP files:
   a. Scans the tape file in order to make a tape map.
   b. Processes each CMS file in the tape file individually.
   c. Looks for CMS files with a filetype of $PTFPART or a PTF-numbered filetype (which has 8 characters, the last 5 of which are whole numbers). If found, the target string is searched for a file with the same filename and filetype. If one is found, the CMS file is not loaded. Otherwise, the file is received on the target disk.
   d. Puts the file in uppercase using the COPYFILE command.

VMFRCTXT This EXEC receives text decks as follows:

a. VMFRCTXT checks to see if a text deck with the same filename and filetype exists in the target string.

   - If a text deck with the same filename and filetype exists on a previous level minidisk or directory, the new deck is not received, and processing for that text deck ends. This prevents cumulative text decks received from a previous service tape from being duplicated in the database.
   - If a text deck with the same filename and filetype exists on the target minidisk or directory, the new deck is received to replace the existing text deck. The existing text deck is probably a text deck shell created by VMFREC, because only one copy of a specific text deck is shipped on the service tape. If the existing text deck is a full text deck because the service tape was received a second time or multiple service tapes were received to the same target, no harm is done by the overlay.

b. VMFRCTXT checks the LTO (last text only) record in the $PPFTEMP file and processes the text deck as follows:

   - If LTO is set to NO, VMFRCTXT loads the text deck to the target.
   - If LTO is set to YES, VMFRCTXT compares the text deck prolog with the prologs of all other text decks that have the same filename and filetype prefix (for example, TXT). VMFRCTXT keeps the text deck with the latest entries and creates text deck shells for the others. If the text deck you receive on the tape does not include all the entries from earlier text decks, VMFRCTXT writes a message into the receive exception log.

c. VMFRCTXT processes the history (prolog) section of all the text decks on the target minidisk or directory:

   - If a source update file does not exist for the text deck (because the part is serviced by replacement), VMFRCTXT creates an update shell on the APPLY minidisk or directory.
   - If a text deck does not exist in the target string, VMFRCTXT creates a text deck shell.

VMFRCCOM This EXEC does the following:

a. Scans the tape file in order to make a tape map.

b. Processes each CMS file in the tape file individually.

c. Looks for CMS files with a filetype of $PTFPART or a PTF-numbered filetype (which has 8 characters, the last 5 of which are whole numbers). If found, the target string is searched for a file with the same filename and filetype. If one is found, the CMS file is not loaded. Otherwise, the file is received on the target disk.

VMFRCAUX This exec loads a tape file containing AUX files from a PUT or COR tape to the target disk in the $PPF. The AUX files are named according to the top (most recent) AUX entry in the file and the 3-character abbreviation of the AUX filetype if it does not already exist on the target string. For example, if the AUX file had the filetype AUXSP and the top entry in it contained the PTF number UV12345 in the third token, the new filetype of the AUX file would be AUC12345. If the file already exists on the target string, the file is erased. VMFRCAUX is called once for every tape file that specifies it as its part handler in the $PPF.

9. After the part-specific EXECs finish processing, control returns to VMFREC with a return code.
10. If a user exec is specified in the USEREXIT tag in the $PPFTEMP file, VMFREC calls the user exec with the parameters VMFREC, CLEAN-UP, and a return code.

Usage Notes

1. The default for VMFREC INFO is MEMOS.
2. If you have disks accessed as read-only that are identified on the MERGE tag, VMFREC will exit. You can override the MERGE tags in the PPF or access the disk as read/write to avoid this.

Return Codes

VMFREC issues the following return codes:

Return Code	Explanation
0	No errors were encountered.
4	A message of type CK, WN, RO, or BD was encountered. Check $VMFREC $ERRLOG.
8	There was a syntax error in the command.
23	Merge processing error.
24	There was an unexpected tag in PPF.
28	The required file was not found.
100	An unrecoverable error was encountered.

VMFREMOV EXEC

The VMFREMOV EXEC procedure removes PTFs applied by the VMFMERGE EXEC procedure.

Do not use this procedure to service any of the base components of VM/XA SP. Use this procedure when applying PTFs to System Network Architecture (SNA) products.

Format

The format of the VMFREMOV EXEC is:

VMFREMOV	*prodid* PTF [*ptfnum*|*] PTFLIST *remlist* CONVERT [*listfilemode*]

Where:

prodid
is the ID of the product.

PTF [*ptfnum*|*]
removes the specified PTF (*ptfnum* is the filename of a PTF) or, if you enter * instead of a *ptfnum*, removes the PTFs listed in a remove list file named *prodid* REMLIST. This file must already exist on a DELTA disk.

PTFLIST ***remlist***
removes the selected PTFs in a remove list file named *remlist* REMLIST. This file must already exist on a DELTA disk.

CONVERT
invokes the tool that creates the **reqby** log file (if one does not already exist), but does **not** remove any changes.

lastfilemode
specifies the filemode of the last DELTA disk. VMFREMOV assumes that the merge disk is always accessed as E and that all other DELTA disks are accessed as consecutive filemodes between F and the *lastfilemode*. If you do not specify *lastfilemode*, then the filemodes for the MERGE and DELTA disks are determined from the information in the VMFPARM file. Use the *lastfilemode* parameter only if you **know** you have the correct disks accessed as the proper modes.

If you wish only to create the **reqby** log and do **not** want to remove any changes, use the CONVERT keyword.

How VMFREMOV Works

VMFREMOV is a service process that removes PTFs applied by VMFMERGE. The EXEC procedure:

1. Obtains data from the merge log and the service control files to build the reqby log if one does not already exist. The reqby log contains a list of all dependent PTFs that must be removed if their requisite PTF is removed.
   - If a service control file for any of the PTFs is missing, then processing continues. However, the reqby log will be incomplete if the missing service control file contains requisites. If any SCFs were missing, processing ends after VMFREMOV completes the build of the reqby log. No PTFs are removed.
2. Checks the merge log to insure that the PTF to be removed is currently merged.
3. Reads the reqby log for the list of dependent PTFs to remove.
4. Removes a PTF (for example, `UV00007`) that may supersede other PTFs (for example, `UV00005`).

   Note: The other PTF (`UV00005`) is no longer superseded and its status remains as it was prior to the merge of the primary PTF (`UV00007`) (that is, merged, superseded, or no status). If the prior status of `UV00005` is no status, then VMFREMOV removes its dependents.
5. Copies, from the DELTA disk to the MERGE disk, each element affected by the PTF being removed if previous service for an element is merged. If the element has not been merged, VMFREMOV erases the element from the MERGE disk.

   The very first line of a copied text deck is always a comment line consisting of the PTF name, and the date and time stamp. Any information on the `:apartext` entry is copied, but the first line is a comment.

   Note: A temporary file (with filetype of OVMFMGLG) is created during this procedure to insure system integrity. This file is erased during normal VMFREMOV processing.
6. Removes the element's entry from the **reqby** log. If an element has other elements that are dependent upon it, VMFREMOV also removes those dependent entries from the **reqby** log.
7. Updates the merge log with the current status of the PTFs. The merged entry is commented out. Another comment is added to the end of the merge log (with a time and date stamp) indicating that the PTF has been removed.

   In the case where a PTF that supersedes another PTF is removed, the **superseded** entry is commented out.

Note: If you remove a change, VMFREMOV has no way of knowing if the change is a requisite of a change in another product. You should make note of any cross-product requisite information during VMFMERGE processing (see page 669) in order to know which changes to manually remove.

Messages

DMSWRM002E **File** *fn ft* [*fm*] **not found.**
DMSWRM002W **File** *fn ft* [*fm*] **not found.**
DMSWRM008E **Device** *vdevno* **invalid or nonexistent.**
DMSWRM017E **Invalid device address** *vdevno.*
DMSWRM520E **Invalid operand:** *operand.*
DMSWRM545E **Missing operands.**
DMSWRM632E **I/O error in EXECIO; RC-***nn* **from** *command* **command.**
DMSWRM649E **Extraneous parameter** *parm.*
DMSWRM653E **Error executing** *command.*
DMSWRM823E **PTF** *name1* **is listed as a dependent of PTF** *name2*, **but PTF** *name1* **is not merged.**
DMSWRM824W *prodid* **VMFREQBY may be incomplete due to a missing SCF.**

DMSWRM856E **Disk address *vdevno* is listed more than once on the DELTA and/or MERGE entry records in the *prodid* VMFPARM file.**
DMSWRM857E **The number of disk addresses on the DELTA entry record cannot exceed nine.**
DMSWRM858E **Unable to find a *tag* entry record in the *fn ft* file.**
DMSWRM859E **The *prodid* VMFPARM file has no disk addresses on the {MERGE|DELTA} entry record.**
DMSWRM860E **Only one {MERGE|DELTA} entry record may appear in the *prodid* VMFPARM file.**
DMSWRM861I **Accessing *disk_type* disk *vdevno* as *fm*.**
DMSWRM862I **Change *name* {has been *action*|is no longer SUPERSEDED by *name*}.**
DMSWRM863E **The MERGE disk *vdevno* must be linked read-write.**
DMSWRM864W **PTF *name* will not be *action* because it is not *status*.**
DMSWRM865I **Processing PTF *name*.**
DMSWRM866W **No PTFs have been *action*.**
DMSWRM867E **Invalid status *status* in *prodid* VMFMGLOG for entry *PTF*.**
DMSWRM874E **Invalid entry found at line *line* in *fn ft*.**
DMSWRM879W **Change name *name* appears more than once in the *fn ft*.**
DMSWRM882E **File *fn ft* [*fm*] [from *name*] not found on any DELTA disks from the disks from the VMFPARM file.**
DMSWRM883W **PTF *name* is not a part of product *prodid* and must be *action* in product *prodid*.**
DMSWRM888E **Error in *name* SCF. No entry for element *fn ft*.**
DMSWRM892E **PTF *name* has not been *action*.**
DMSWRM893W **Incomplete processing, not all [required] PTFs were *action*.**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference*.

VMFSETUP EXEC

The VMFSETUP EXEC uses the minidisk assignment section of the product parameter file to establish the correct minidisk access order for the component being serviced. It can also be used to release the accesses it established and restore the previous access order. VMFSETUP is invoked by VMFAPPLY, VMFHASM, and VMFBLD. You can also invoke it from the command line.

Format

The format of the VMFSETUP EXEC is:

```
VMSETUP   prodid compname  [PPFTEMP]  [(options... [)]]
          options:
          [ACCESS   [ASM]|RESTORE]
                    [APP]
                    [BLD]
                    [REC]
                    [ALL]

          [RETAIN m m m ...]
```

Where:

prodid
is the product number of the product to be serviced. For VM/XA SP, the product identifier is 56643089.

compname
is the name of the component to be serviced. The names for the VM/XA System Product Release 2.1 components are CP, CMS, GCS, and DV.

PPFTEMP
indicates that the caller has created a $PPFTEMP file that contains only one component (keyword). If not specified, VMFSETUP calls VMFOVER to create a $PPFTEMP file based on the *prodid* $PPF file and any existing override file.

***options*:**

ACCESS
indicates that a new minidisk access order should be set up. ACCESS is the default.

Note: If a minidisk is in access mode read-write, and you have specified that minidisk as a read-only disk in your prodcct parameter file, VMFSETUP releases the minidisk and re-accesses it as a read-only disk.

RESTORE
indicates that the accesses should be released and the previous access order restored.

REC
sets up the access order for VMFREC.

APP
sets up the access order for VMFAPPLY. Whenever one of these EXECs invokes VMSETUP, it specifies APP.

ASM
sets up the access order for VMFHASM. Whenever VMFHASM invokes VMSETUP, it specifies ASM. ASM is the default option.

BLD
sets up the access order for VMFBLD. Whenever VMFBLD invokes VMSETUP, it specifies BLD.

ALL
sets up all the accesses specified in the minidisk assignment section of the product parameter file.

RETAIN ***m m m...***
each *m* indicates a filemode that is not to be used by VMFSETUP.

How VMFSETUP Works

If VMFSETUP is invoked from the command line rather than from another EXEC, VMFSETUP invokes VMFOVER.

If the file named *productid* $SETUP exists, the current accesses are validated against the release section of *productid* $SETUP. If the accesses do not match, the user is notified and given the option of terminating or releasing everything but the A-disk.

If the accesses match, the original accesses saved in the restore section of the file with the filename of the product name and a filetype of $SETUP are restored. If the RESTORE option was specified, VMFSETUP processing is complete.

If the ACCESS option was specified or chosen by default, VMFSETUP locates the minidisk assignment section (:MDA.) in the chosen component section of the product parameter file. VMFSETUP then saves the current minidisk access order in the restore section of *productid* $SETUP.

Minidisks are accessed according to the process options specified. Table 20 shows the minidisk access order for each option:

Table 20. Minidisk Accesses Established by VMFSETUP EXEC

	REC	APP	ASM	BLD	ALL
TASK	X	X	X	X	X
LOCAL*xxx*–LOCAL*yyy*			X	X	X
APPLY*xxx*–APPLY*yyy*	X	X	X	X	X
DELTA*xxx*–DELTA*yyy*	X	X	X	X	X
LOCAL*xxx*–LOCAL*yyy*		X			
BASE*xxx*–BASE*yyy*	X	X	X	X	X
BUILD*xxx*–BUILD*yyy*	X	X	X	X	X
SYSTEM	X	X	X	X	X

Finally, VMFSETUP does a QUERY SEARCH, displays the results on the screen, and stores the results in the log. In addition, these new accesses are saved in the release section of the file named *productid* $SETUP in the form needed to validate before a subsequent release.

Note: If a minidisk is in access mode read-write, and you have specified that minidisk as a read-only disk in your prodcct parameter file, VMFSETUP releases the minidisk and re-access it as a read-only disk.

VMFVIEW EXEC

The VMFVIEW EXEC invokes XEDIT to allow you to view the exception logs. Using the VMFVIEW EXEC's PF key assignments, you can:

- View all the messages of a specific type
- View all the messages of a specific number
- View the HELP screen for a particular message
- Move backwards and forwards through the displayed exception log.

The PF key assignments that the VMFVIEW EXEC uses are defined in a tailorable profile called VMFVIEW$ PROFILE. See "The VMFVIEW Profile File" on page 704for a description of this profile.

Format

The format of the VMFVIEW EXEC is:

VMFVIEW	? | *errlog* [(LAST | ALL] *errlog:* [Receive|VMFRec |$VMFRec Apply \|VMFApply\|$VMFApp Build |VMFBld |$VMFBld|BLD]

Where:

?
causes a VMFVIEW help screen to be displayed.

errlog
is the exception log you want to view. Specify:

- **Receive, VMFRec**, or **$VMFRec** to view the receive exception log
- **Apply, VMFApply**, or **$VMFApp** to view the apply exception log
- **Build, VMFBld, $VMFBld**, or **BLD** to view the build exception log.

LAST
specifies that only messages from the most recent run in the exception log will be displayed. **LAST** is the default for VMFVIEW.

ALL
specifies that messages from all runs in the exception log will be displayed.

Default PF Key Assignments

The default PF keys for the VMFVIEW EXEC are:

PF01/13 - Help	Display a HELP screen (see Usage Note 3)
PF02/14 - All	Display ALL messages (see Usage Note 4)
PF03/15 - Quit	Exit VMFVIEW
PF04/16 - Exception	Display CK:, MS:, RQ:, SV:, and WN: messages
PF05/17 - Status	Display ST: messages
PF06/18 - Build	Display BD: messages
PF07/19 - Backward	Display previous screen of messages
PF08/20 - Forward	Display following screen of messages
PF09/21 - OutCompRq	Display RO: messages
PF10/22 - Non-Stat	Display all messages except ST: messages
PF11/23 - Requisite	Display RQ: messages
PF12/24 - Severe	Display SV: messages.

Message Headers

The message headers that appear in the exception logs are:

BD These are "build" messages that give information needed to rebuild the parts affected by the application of service.

CK These are items that you **must** check. These messages contain information that may require user action.

MS These are "mismatch" messages that indicate conflicting control information that **must** be investigated and corrected.

RO These are "requisite out of component" messages. They indicate that PTFs **must** be applied to another component.

RQ These are "requisite" messages that indicate that there are PTFs (in the same component) that are missing and **must** be applied.

ST These are "status" messages. These message give useful information but require no further action.

SV These are "severe" messages. These indicate problems that resulted in the termination of a process. These problems **must** be investigated and corrected.

WN These are "warning" messages that **must** be investigated. They indicate situations that may or may not be a real problem. It is up to the user to decide after investigating the cause of the message.

Usage Notes:

1. PF keys are defined by the file VMFVIEW$ PROFILE. You can change the PF key assignments by editing this file. Instructions for tailoring the PF keys appear at the beginning of this file.
2. The initial set of exception log messages displayed by VMFVIEW is defined in the file VMFVIEW$ PROFILE. The default is the "exception" category of messages.
3. The Help PF key performs one of two functions, depending on the position of the cursor:
   a. When the cursor is on the command line or on a line with no message number, the VMFVIEW help screen will be displayed
   b. When the cursor is on a line that contains a message number, the help screen for the corresponding CMS message number will be displayed. If the requested help screen cannot be found, a message is displayed.
4. The "All" PF key performs one of two functions, depending on the position of the cursor:
   a. When the cursor is on the command line or on a line with no message number, the "All" key causes all of the messages in the log to be displayed
   b. When the cursor is on a line that contains a message number, the "All" key causes all of the messages in the log with this message number to be displayed. Note that these keys are still qualified by the LAST|ALL command line option of VMFVIEW.
5. To gather information about messages, you can issue commands such as FILELIST and XEDIT from the command line.
6. You may find it useful to put a subset of messages into a separate CMS file. To do this you can use the XEDIT PUT command. An example follows:

   If you want to put all BD: (build) messages in a file to process separately, press the appropriate PF key to isolate this subset of messages. (The default is PF06/18.) Then enter the following command on the XEDIT command line:

   ```
   PUT * BUILD
   ```

   This creates a CMS file with the name BUILD $ERRLOG. The PUT command changes the current line on the XEDIT screen. You can enter TOP on the XEDIT command line line to make the current line the top line of the file or press another PF key to view a new subset of messages.
7. You may find it useful to add the invocation of VMFVIEW to the service user exits. To do this, include the following commands in your user exit:

   ```
   parse upper arg execname flag retcode
   if function = 'CLEAN-UP' then do
     'EXEC VMFVIEW' execname
   ```
8. If, while viewing messages from the LAST run, you wish to check messages in previous runs, use the "ALL" XEDIT macro. This macro resets the "SET SELECT" and "SET DISPLAY" XEDIT sub-commands that are used by VMFVIEW. To return to a view of only the LAST run, press one of the PF keys that is set to select a subset of messages.

The VMFVIEW Profile File

VMFVIEW uses a profile file called VMFVIEW$ PROFILE. You can tailor this file to suit your needs. Figure 53 on page 705 shows the VMFVIEW$ PROFILE file supplied by IBM.

```
************************************************************
* THIS PRODUCT CONTAINS RESTRICTED MATERIALS OF IBM.       *
* COPYRIGHT - 5664-308                                     *
* (C) COPYRIGHT IBM CORPORATION - 1989                     *
*           LICENSED MATERIAL - PROPERTY OF IBM            *
************************************************************
*                                                          *
* Name: VMFVIEW$ PROFILE                                   *
*                                                          *
* Function: Tailorable profile for VMFVIEW.                *
*                                                          *
* Syntax of Entries in the Profile:                        *
*                                                          *
*   * cmt                                                  *
*   VMFVIEW PFnn ( text ) key * cmt                        *
*   VMFVIEW INIT ( text ) key * cmt                        *
*   COMMAND xcmd                                           *
*   MACRO   xmac                                           *
*                                                          *
*   where  cmt   is a comment.                             *
*          nn    is a number between 1 and 24 inclusive.   *
*          text  is the text to describe a PF key.  If     *
*                necessary, this field is truncated to 9   *
*                characters or padded with blanks.         *
*          key   is one of the following:                  *
*                'HELP'     - display help screens         *
*                'ALL'      - display all messages         *
*                'QUIT'     - exit from VMFVIEW            *
*                'BACKWARD' - display previous screen      *
*                'FORWARD'  - display following screen     *
*                'xx:<,yy:<,zz:<...>>> with NO spaces      *
*                           - display messages with the    *
*                             listed headers               *
*                '¬xx:<,¬yy:<,¬zz:<...>>> with NO spaces   *
*                           - display all messages except  *
*                             those with the listed headers*
*          xcmd  is an Xedit command.                      *
*          xmac  is an Xedit macro.                        *
*                                                          *
* Usage Notes:                                             *
*                                                          *
* 1) Use an asterisk '*' to 'comment out' old entries      *
*    instead of deleting them or changing them directly.   *
*    This will make it easier to recreate the original     *
*    definitions.                                          *
*                                                          *
* 2) The valid headers are:                                *
*      BD:, CK:, MS:, RO:, RQ:, ST:, SV:, WN:              *
*                                                          *
* 3) It is recommended that PF keys 13-24 duplicate PF     *
*    keys 1-12 because only PF keys 1-12 are displayed     *
*    in VMFVIEW.  If a PF key is not defined the Xedit     *
*    default is used.                                      *
```

Figure 53 (Part 1 of 2). VMFVIEW$ PROFILE

```
* 4) The entries in this profile are processed in the       *
*    same order in which they appear.  This means that      *
*    you can override previous settings.                    *
*                                                           *
* 5) You may use the Xedit command SET PFxx.  If you do,    *
*    the text describing that PF key will be blank.         *
*                                                           *
* 6) Xedit commands and macros must have their correct      *
*    syntax.                                                *
*                                                           *
* 7) The 3 lines on the top of the screen and the 2 lines   *
*    above the bottom line of the screen are reserved for   *
*    VMFVIEW.                                               *
*                                                           *
*************************************************************
VMFVIEW INIT (Exception) CK:,MS:,RQ:,SV:,WN:
VMFVIEW PF01 (Help     ) HELP
VMFVIEW PF02 (All      ) ALL
VMFVIEW PF03 (Quit     ) QUIT
VMFVIEW PF04 (Exception) CK:,MS:,RQ:,SV:,WN:
VMFVIEW PF05 (Status   ) ST:
VMFVIEW PF06 (Build    ) BD:
VMFVIEW PF07 (Backward ) BACKWARD
VMFVIEW PF08 (Forward  ) FORWARD
VMFVIEW PF09 (OutCompRq) RO:
VMFVIEW PF10 (Non-Stat ) ¬ST:
VMFVIEW PF11 (Requisite) RQ:
VMFVIEW PF12 (Severe   ) SV:
VMFVIEW PF13 (Help     ) HELP
VMFVIEW PF14 (All      ) ALL
VMFVIEW PF15 (Quit     ) QUIT
VMFVIEW PF16 (Exception) CK:,MS:,RQ:,SV:,WN:
VMFVIEW PF17 (Status   ) ST:
VMFVIEW PF18 (Build    ) BD:
VMFVIEW PF19 (Backward ) BACKWARD
VMFVIEW PF20 (Forward  ) FORWARD
VMFVIEW PF21 (OutCompRq) RO:
VMFVIEW PF22 (Non-Stat ) ¬ST:
VMFVIEW PF23 (Requisite) RQ:
VMFVIEW PF24 (Severe   ) SV:
*************************************************************
COMMAND SET ENTER IGNORE COMMAND CURSOR HOME PRIORITY 30
COMMAND SET CMDLINE BOTTOM
COMMAND SET CURLINE ON 4
COMMAND SET MSGLINE ON 2 2 OVERLAY
COMMAND SET MSGMODE ON LONG
COMMAND SET COLOR MSGLINE RED
COMMAND SET COLOR CURLINE WHITE
COMMAND SET COLOR FILEAREA YELLOW
COMMAND SET SCALE OFF
COMMAND SET ETMODE OFF
COMMAND SET SHADOW OFF
COMMAND SET PREFIX OFF
```

Figure 53 (Part 2 of 2). VMFVIEW$ PROFILE

VMFZAP EXEC

Use the VMFZAP EXEC procedure to apply ZAPs and to maintain a record of them in the ZAP log.

Do not use this procedure to service any of the base components of VM/XA System Product. Use this procedure when applying PTFs to System Network Architecture (SNA) products.

This EXEC uses the BASE disk, MERGE disk, and ZAP disk as inputs and produces an updated ZAP disk as output.

Format

The format of the VMFZAP EXEC is:

VMFZAP	*prodid*

Where:

prodid
is the ID of the product you want to ZAP.

When you issue VMFZAP, you must specify the name of the product you want to ZAP. VMFZAP accesses the disks using filemode letters E–N. VMFZAP accesses the MERGE disk ahead of the BASE disk. When VMFZAP processing stops, your search hierarchy is restored.

In order to use VMFZAP, you MUST have an A-disk accessed in read/write mode. This disk MUST NOT be the ZAP disk, MERGE disk, or BASE disk. That is, the virtual address of your A-disk MUST NOT appear on the ZAP, MERGE, or BASE records of your VMFPARM file.

How VMFZAP Works

VMFZAP:

1. Uses a parameter file to determine the virtual addresses of the ZAP, MERGE, and BASE disks.
2. Reads the ZAP log and builds a list of TEXT filenames with ZAPs applied to them. Then it:
   - Erases all TEXT files in this list from the ZAP disk
   - Applies ZAPs to the first version of the TEXT file found among the other disks
3. Erases the ZAP log.
4. Reads the merge log and builds a list of ZAPs that are currently superseded.
5. Reads the ZAP List for the names of all ZAPs you want to apply.
6. Checks each ZAP name to see if it is superseded. If it is not superseded, then VMFZAP reads the control file for that ZAP.

   Control files for a product ZAP may contain information for ZAPping more than one text file. VMFZAP separates this information by text filename and processes the ZAP of each text file in the order they are listed in the control file.

   Checks each text filename to see if it resides on some disk other than the ZAP disk. It is an error if the text file resides on the ZAP disk already.

If the text file exists, VMFZAP writes a temporary file called $$VMFZAP ZAP to the ZAP disk containing the ZAP control records for the current text file.

Copies the text file to the ZAP disk under a temporary name of the format VMF$T*n* TEXT, where *n* is a number determined by how many text files are affected by the ZAP currently being processed. When this temporary file has been successfully ZAPped it is renamed to its original name on the ZAP disk.

7. Calls ZAPTEXT passing the $$VMFZAP name and the VMF$T*n* filename to be ZAPped.

   Refer to "ZAPTEXT EXEC" on page 720 for more information about the ZAPTEXT command.

8. Updates the ZAP log with the information about the TEXT file that was just ZAPped.
9. Restores your disk search hierarchy once all ZAPs in the ZAPLIST have been processed.

Messages

DMSWZP002E **File *fn ft* [*fm*] not found.**
DMSWZP008E **Device *vdev* invalid or nonexistent.**
DMSWZP017E **Invalid device address *vdev*.**
DMSWZP520E **Invalid operand : *operand*.**
DMSWZP545E **Missing operands.**
DMSWZP649E **Extraneous parameter *parm*.**
DMSWZP653E **Error executing *command*.**
DMSWZP856E **Disk address *vdev* is listed more than once on the {BASE, ZAP | DELTA} and/or MERGE entry records in the *prodid* VMFPARM file.**
DMSWZP858E **Unable to find a *tag* entry record in the *fn ft* file.**
DMSWMG859E **The *prodid* VMFPARM file has no disk addresses on the {BASE | MERGE | ZAP} entry record.**
DMSWMG860E **Only one {BASE | MERGE | ZAP} entry record may appear in the *prodid* VMFPARM file.**
DMSWZP861I **Accessing *disk_type* disk *vdev* as *mode*.**
DMSWZP862I **ZAP *name* has been *action*.**
DMSWZP863E **The ZAP disk *vdev* must be linked read-write.**
DMSWZP864W **ZAP *name* will not be *action* because it already is *status*.**
DMSWZP865I **Processing ZAP *name*.**
DMSWZP874E **Invalid entry found at line *line* in *fn ft*.**
DMSWZP875E **File *fn ft* [*fm*] not found on any disks from the VMFPARM file.**
DMSWZP876E **The total number of disk addresses on the BASE and MERGE entry records cannot exceed nine.**
DMSWZP877W ***fn* TEXT was previously zapped but was not found on the ZAP disk.**
DMSWZP878E ***prodid* ZAPLIST does not contain any unsuperseded zap names. No zaps will be applied.**
DMSWZP879W **ZAP name *name* appears more than once in the *fn ft* [It will only be applied once.]**
DMSWZP880E **Error in ZAPTEXT while processing *fn* TEXT. Textfiles affected by *filename2* ZAP will not be saved on the ZAP disk.**
DMSWZP881E ***fn* TEXT was found on the zap disk but was not zapped during the VMFZAP run. This file should not be on the ZAP disk.**
DMSWZP885I **File *prodid* VMFZPLOG not found on the ZAP disk. No text files will be removed from the ZAP disk.**
DMSWZP886E **Filename *name* from the *fn ft* [*fm*] file is longer than 8 characters.**
DMSWZP887E **Record number *number* from the *fn ft* [*fm*] file is longer than 80 bytes.**

For a complete explanation of each message, refer to *VM/XA SP System Messages and Codes Reference*.

VSEVSAM EXEC

Use the VSEVSAM EXEC to obtain VSE/VSAM Assembler Language macros from the Licensed Optional Machine Readable Materials tape. The VSEVSAM EXEC creates the VSEVSAM MACLIB for you. Once the MACLIB is created, it contains all of the VSE/VSAM assembler language macros, and the following VSE macros: CDLOAD, CLOSE, CLOSER, GET, OPEN, OPENR, and PUT.

The format of the VSEVSAM EXEC is:

VSEVSAM	

Example of Using VSEVSAM

Before invoking the VSEVSAM EXEC, complete the following:

1. Mount the Licensed Optional Machine Readable Materials tape at virtual address 181.
2. Load the seven VSE macros from the Product Tape to MAINT 393 or a minidisk of your choice. (As long as the macros are available when VSEVSAM is invoked, the actual minidisk used is not critical.)

   Note: Because the seven VSE macros (CDLOAD, CLOSE, CLOSER, GET, OPEN, OPENR, and PUT) will be loaded into the MACLIB, they can be erased from the disk after the MACLIB is created.

To invoke VSEVSAM EXEC, enter:

```
vsevsam ■
```

Once invoked, VSEVSAM EXEC prompts you for information. For example, the system responds:

```
DMSWVV797I "QUIT" may be entered in response to any query to
           end processing.

DMSWVV788R Are the macros to be read from tape or are
           they already on disk?  Reply  TAPE or DISK.
           If a default of TAPE is to be used, press "ENTER."
```

For this example, you read the macros from disk. Enter:

```
disk ■
```

The system responds:

```
DMSWVV790R If the default library name of "VSEVSAM" is to be
           used, press "ENTER."  Else, enter the name to be
           used for the library.
```

You want to call the library VSAMMACS, so enter:

```
vsammacs ■
```

The system responds:

```
DMSWVV791I The library name will be "VSAMMACS."
           Press "ENTER" to continue, else enter
           "QUIT" or the name to be used for the library.
```

Because VSAMMACS is the name we want to call the library, press:

■

The system responds:

```
DMSWVV808R Macro library libname will be erased.
           Press "ENTER" to continue or type "QUIT" to exit.
```

Because we want to erase old versions of the library that might be on our A-disk, press:

■

The system responds:

```
DMSWVV793I Maclib generation completed.

DMSWVV792R Are the macros to be erased from disk?
           Reply (YES | NO).  Press "ENTER" for default
           of "YES."
```

We want to erase the macros from disk, so press:

■

The system responds:

```
DMSWVV802I Macros erased - VSEVSAM processing complete.
Ready;
```

Messages

DMSWVV788R **Are the macros to be read from tape or are they already on disk? Reply TAPE or DISK. If a default of TAPE is to be used, press "ENTER".**

DMSWVV789W **Invalid response.**

DMSWVV790R **If the default library name of "VSEVSAM" is to be used, press "ENTER". Else, enter the name to be used for the library.**

DMSWVV791I **The library name will be *libname*. Press "ENTER" to continue, else enter "QUIT" or the name to be used for the library.**

DMSWVV792R **Are the macros to be erased from disk? Reply (YES | NO). Press "ENTER" for default of "YES".**

DMSWVV793I **Maclib generation completed.**

DMSWVV794E **Error in maclib generation.**

DMSWVV795E **Error reading macros from tape.**

DMSWVV796E **Error reading from "VSEVSAM SCAN" file.**

DMSWVV797I **"QUIT" may be entered in response to any query to end processing.**

DMSWVV798R **The VSE/VSAM Optional Source Statement Library tape must be mounted as virtual 181. If it is not, enter "QUIT" here and have the tape mounted. Else, press "ENTER" to continue.**

DMSWVV799E **Error reading from "VSEVSAM SCAN" file - all macros may not be erased.**

DMSWVV800E **One of the files needed for maclib generation is missing.**

DMSWVV801I **Arguments entered are ignored.**

DMSWVV802I **Macros erased - VSEVSAM processing complete.**

DMSWVV808R **Macro library *libname* will be erased. Press "ENTER" to continue or type "QUIT" to exit.**

DMSWVV809E **Error copying "VSEVSAM SCAN" file from S-disk to A-disk.**

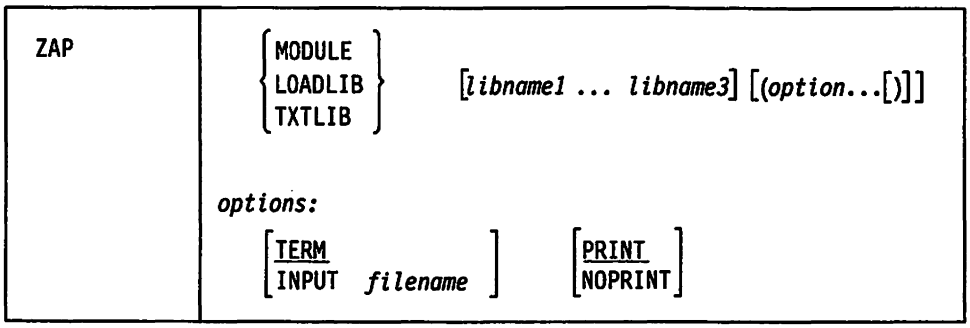
ZAP MODULE

ZAP is a CMS command that changes or dumps MODULE, LOADLIB, or TXTLIB files. It may be used to change either fixed or variable length MODULE files. It is for use by system support personnel only.

Input control records control ZAP processing. They can be submitted either from the terminal or from a disk file. Using the VER and REP control records, you can verify and replace data or instructions in a control section (CSECT). Using the DUMP control record, you can dump all or part of a CSECT, an entire member of a LOADLIB or TXTLIB file, or an entire module of a MODULE file.

Format

The format of the ZAP MODULE is:



Where:

MODULE
LOADLIB
TXTLIB
indicates the type of file that you want to change or dump.

libname
is the filename of the library containing the member you want to change or dump. You can specify one to three library names. This operand is valid only for LOADLIB and TXTLIB files.

options:

TERM [**PRINT** | **NOPRINT**]

indicates that input to the ZAP service program is submitted through the terminal. If you specify TERM, the prompting message ENTER: is issued, and you can then enter input control records up to 80 characters long. If you specify PRINT with TERM, all output prints on the printer, but only error messages display at the terminal. If you specify NOPRINT with TERM, nothing prints on the printer, and all output except control records displays at the terminal. See Table 21 on page 713.

INPUT ***filename*** [**PRINT** | **NOPRINT**]

specifies that input is submitted from a disk file called *filename*. This file must have a filetype of ZAP, and must be a fixed 80-byte sequential file residing on any accessible device. If you specify PRINT with INPUT *filename*, all output produced by the ZAP service program prints on the printer. In addition, commands in error, control records in error, and error messages display at the terminal. If you specify NOPRINT with INPUT *filename*, nothing prints on the printer, and all output displays at the terminal. See Table 21 on page 713.

Table 21. Valid Options and Their Output		
	PRINT	**NOPRINT**
INPUT	Everything prints on the printer. Commands in error, control records in error, and error messages display on the terminal.	Nothing prints on the printer. Everything displays on the terminal.
TERM	Everything prints on the Printer. Error messages display on the terminal.	Nothing prints on the printer. Everything except control records displays on the terminal.

Input Control Records

There are eight types of ZAP control records:

- DUMP
- NAME
- BASE
- VER or VERIFY
- REP
- LOG
- COMMENT
- END.

The ZAP program can accept only 80 characters of data for each control record. ZAP control records are free-form and need not start in position one of the record. Separate all information by one or more blanks. All address fields including disp (displacement) fields in VER and REP control records must contain an even number of hexadecimal digits, to a maximum of six digits (X'OD', X'02C8', X'014318'). Data fields in VER and REP control records must also contain an even number of hexadecimal digits.

If you want, you can separate the data anywhere by commas (for example, 83256482 or 8325,6482). The commas have no effect on the operation.

Note: Do not use blank spaces as separators *within* data fields.

The program sets the NOGO switch on if it finds a control record in error. A file cannot be changed if the NOGO switch is turned on. The next valid NAME record turns the NOGO switch off. This means that if the control record is the NAME record, all succeeding records are ignored until the next NAME, DUMP, or END record. For any other error, only REP control records that follow are ignored.

DUMP Control Record

The DUMP control record allows you to dump a portion or all of a specified control section, or the complete member or module. The format of the output of the dump is hexadecimal with an EBCDIC translation of the hexadecimal data.

The DUMP control record is optional and resets the NOGO switch off. The DUMP control record must not immediately precede a BASE, VER, or REP control record. A NAME control record must precede the BASE, VER, and REP control records (if any) that follow a DUMP control record.

Format:

Where:

membername
is the name of the member you want to dump, or the member that contains the CSECT(s) you want to dump. This member must be in one of the libraries you specify on the ZAP command.

For a CMS TXTLIB, the format of the dump control record requires that you specify both *membername* and *CSECTname*. Because the library directory does not contain member names, any word may be used to replace *membername*. The program searches for only the second name following the dump operand; therefore, the second name must be *CSECTname*.

modulename
is the name of the module you want to dump, or the module that contains the CSECTs you want to dump. If you specify a module that has no loader table, the program dumps the entire module.

CSECTname
is the name of the control section that you want to dump. If you do not specify *CSECTname*, the program dumps only the first CSECT. *CSECTname* is required for CMS TXTLIBs, but is optional for OS TXTLIBs, LOADLIBs, and MODULE files. (See the discussion of CSECTname in "NAME Control Record" on page 715). You must not specify *CSECTname* for a module created with the NOMAP option.

startaddress
is the location within the specified CSECT where you want the dump to begin. This must be two, four, or six hexadecimal digits. The start address is the displacement from the beginning of the CSECT. For example, to start dumping at address 08 in a CSECT that begins at location X'400', you specify start address X'08', not X'0408'.

endaddress
is the last address you want to dump. This must be two, four, or six hexadecimal digits. If you do not specify *endaddress*, the program dumps from *startaddress* to the end of the CSECT. Note that start and end addresses apply only when you specify *CSECTname*.

ALL
tells the program to dump all CSECTs within the member or module you specify. You can specify ALL for MODULE files, LOADLIBs, and OS TXTLIBs, but not for CMS TXTLIBs. To dump all the CSECTs in a member of a CMS TXTLIB, you must issue a separate DUMP control record for each CSECT.

Usage Notes:

1. Displacements listed in the dump output for a module file are calculated from the beginning location of the module. Therefore, the addresses in the output may differ from the displacements within a CSECT.
2. If a DUMP control record references a TXTLIB file that contains ORG statements causing more than one occurrence of an address, data found at the first occurrence is displayed, but any subsequent redefinition of data for the same location is ignored.

NAME Control Record

The NAME control record specifies the member or module and CSECT that contain the data you want the ZAP operation to verify or replace. The NAME control record must precede the BASE, VER, and REP control records. If it does not, the program sets the NOGO switch on.

Format:

```
NAME    {membername}   [csectname]
        {modulename}
```

Where:

membername
is the member that you want to search for the desired CSECT.

modulename
is the module that you want to search for the desired CSECT.

CSECTname
is the name of the control section you want to change.

Usage Notes:

1. You must specify *CSECTname* if the CSECT you want to change is in a CMS TXTLIB (that is, a TXTLIB created by the TXTLIB command from CMS TEXT decks that do not have a NAME card following the END card). The directory of a CMS TXTLIB contains only CSECT names and no member names. Select a word to replace *membername* as the first entry following the NAME operand in the NAME statement for a CMS TXTLIB.

   Note: The word you specify for the *membername* for a CMS TXTLIB should be a meaningful name. The filename of the LOG control record is determined by the *membername* or *modulename* you specify in the NAME control record.

2. The CSECT name you specify in the NAME record is compared with CSECT names in the directory. If the CSECTs match and no member name match is found, the member selected is the one that contains the CSECT name.

3. *CSECTname* is optional if the CSECT you want to change is a LOADLIB or an OS TXTLIB (that is, a TXTLIB created by the TXTLIB command from CMS TEXT decks that have a NAME card after the END card). The dictionaries of the specified libraries are searched for the member name and the member is then searched for the CSECT name, if you specified one. If you do not specify *CSECTname* for a LOADLIB or an OS TXTLIB, the program uses the first control section.

4. *CSECTname* is optional for a MODULE file. The module named in the NAME control record is found and, if you specify *CSECTname*, the first record is read to determine the number of records in the module and the availability of a loader table, which the program can then search for *CSECTname*. If you do not specify *CSECTname*, the program uses the beginning location of the module. You cannot specify *CSECTname* if the module was created with the NOMAP option.

BASE Control Record

The BASE control record adjusts displacement values for subsequent VER or REP control records for a CSECT whose starting address is not location zero in an assembly listing.

The BASE control record is optional. See the discussion under "VER or VERIFY Control Record." If you specify the BASE control record, it must follow the NAME record, but it need not follow the NAME record immediately. For example, you could have the following sequence of control records: NAME, VER, REP, BASE, VER, REP.

Format

```
BASE          address
```

Where:

address
is the starting address of the CSECT. It must be two, four, or six hexadecimal digits.

Usage Note: If you do not specify a CSECTname in the NAME control record, you cannot specify any BASE value other than 00.

Example: For a CSECT starting at location X'400', you specify BASE 0400 in the BASE control record. If a subsequent VER card requests verification of location X'0408', BASE 0400 is subtracted from X'0408', and the program verifies location X'08' in the CSECT. This example applies if you specify TXTLIB, LOADLIB, or MODULE and the module map is present.

However, if no module map is present for a MODULE file (that is, the module was generated with the NOMAP option), then all operations are done as if the BASE address is location X'0'. For example, if you specify a BASE of X'400' and the address you want to look at or change in X'408', then you must specify X'08' and not X'408' in the REP and VER control records. The address in this case is from the start of the module.

VER or VERIFY Control Record

The VER control record requests verification of instructions or data within a CSECT. If the verification fails, the program does not do a subsequent REP operation until it finds another NAME control record.

The VER control record is optional. More than one VER record can follow a single NAME record.

Format:

```
{VERIFY}        disp        data
{VER   }
```

Where:

disp
is the displacement from the start of the CSECT containing the data to be inspected, if you did not submit a BASE control record for this CSECT. *disp* can also be the actual location of the data to be inspected, if you did submit a BASE control record. *disp* must be two, four, or six hexadecimal digits. This displacement does not have to be aligned on a fullword boundary. If this displacement value is outside the limits of the CSECT specified by the preceding NAME control record, the VERIFY control record is rejected.

data
is the data against which the data in the CSECT is compared. This must be an even number of hexadecimal digits.

Usage Note: If the VER control statement references data in a TXTLIB file that is later redefined by ORG statements, only the first data definition is verified.

Example: If the location you want to verify is X'3CC', and the CSECT begins at location X'2B0', you can enter:

```
base 02B0 ■
ver 03CC data ■
```

or you can omit the BASE control record, subtract the CSECT start address from the address of the data, and enter:

```
ver 011C data ■
```

REP Control Record

The REP control record changes instructions or data at the specified location within the CSECT that you specify in a preceding NAME control record. The data specified in the REP control record replaces the data at the CSECT location specified by the disp operand. This replacement is on a one-for-one basis; that is, one byte of data defined in the control record replaces one byte of data at the location that you specified. If the replacement fails, the program does not do additional REP operations until it finds another NAME control record.

The REP control record is optional. More than one REP record can follow a single NAME record.

Format:

```
REP             disp   data
```

Where:

disp
is the displacement from the start of the CSECT of the data you want to replace (if you did not submit a BASE control record for this CSECT). *disp* can also be the actual location of the data if you did submit a BASE control record. *disp* must be two, four, or six hexadecimal digits. This displacement need not address a fullword boundary. If this displacement value is outside the limits of the CSECT being changed, the program does not do the replacement operation.

data
is the data that is to replace the data in the CSECT. This must be an even number of hexadecimal digits.

Usage Notes:

1. Although you do not have to verify a location before replacing data, you should do so to make sure that the data being changed is what you expect it to be.
2. If the REP control statement references data in a TXTLIB file that is later redefined by ORG statements, the replacement of data takes place at the first occurrence of the data address of the TXTLIB member.

Example: If the location you want to replace is X'3CC', and the CSECT begins at location X'2B0', you can enter:

```
base 02B0 ■
rep 03CC data ■
```

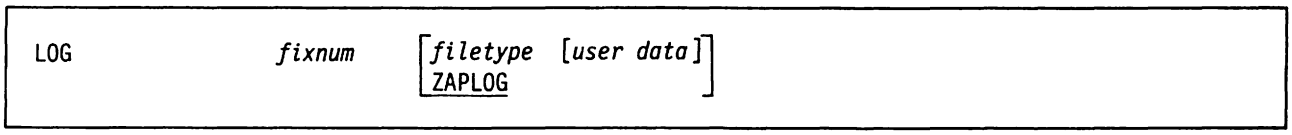
or you can omit the BASE control record, subtract the CSECT start address from the address of the data, and enter:

```
rep 011C data
```

LOG Control Record

The LOG control record lets you specify, after you apply a fix, a unique fix number, which is recorded in a log file for the module or member. The filename of the log file is the same as *membername* or *modulename* in the NAME control record.

Format:



Where:

fixnum
specifies the number associated with the fix. Its length may vary from 1-to 8-alphameric characters.

filetype
specifies the filetype of the log. The default is ZAPLOG.

user data
specifies any data that you want to enter into the log. If you specify *user data*, you must specify *filetype*.

Usage Notes:

1. The LOG control record is optional and is allowed only if valid NAME and REP control records are found. The filename is obtained by the log routine from *modulename* or *membername* in the NAME control record. However, if no LOG control record is found, a dummy log record is written at the end of the user's valid REPs.
2. Log multiple names by including a LOG control record after each name. If the LOG record is not included after each name, error message DMSZAP070E results. Processing continues after the error messages occur.
3. The LOG record is 80 bytes in length and contains the following information:
   - Columns 1 – 63 contain the fixnum and, if specified, the filetype and user data
   - Columns 64 – 80 contain the date and time of the ZAP.

COMMENT Control Record

The ZAP program ignores COMMENT control records. If the PRINT option is in effect, the program prints the comments.

Format:

```
* comment
```

There must be at least one blank following the asterisk (*) before you enter the text.

END Control Record

The END control record ends ZAP processing. The END record is required and must be the last control record for input from the console.

Format:

```
END
```

Special Considerations for Using the ZAP Service Program

Before using the ZAP command against MODULE files, you can use the MODMAP command to determine whether a module map exists and what it contains.

When a ZAP input file has more than one pair of VER and REP control records, and a VER control record (other than the first) fails, you must remove the records prior to the failing record and correct the error before you issue the ZAP command again. Otherwise, the file being changed returns to its original status.

The REP control record cannot be used to place data in an undefined area such as a Define Storage area. If any part of a data field specified in a pair of VER and REP control records is an undefined area, the system displays warning message DMSZAP248W, and no data replacement occurs. If you do not issue a VER control record prior to the REP control record, some change to data may result. User-defined data may be inserted in undefined areas of text files by using the REP statement described under the LOAD command.

If the file to be dumped contains undefined areas (such as a DS or ORG statement in a TXTLIB member), the hexadecimal portion of the dump contains blanks to indicate that the corresponding positions are undefined.

VER and REP control words can be used to change TXTLIB members produced by FORTRAN compilers that store the length of the compiled text in the END card rather than in the ESD card. However, if a member of this type contains multiple CSECTs, only the first CSECT can be changed by the ZAP program.

The TXT records should be in ascending address order. If ZAP finds a TXT record with an address higher than the specified address, it stops scanning for the specified address. This means that ZAP control records affect only the data at the first occurrence of the address.

When applying ZAPs to a text deck created by a compiler, be aware that some compilers, such as FORTRAN, may generate a text deck in which the TXT records are not in ascending address order.

ZAPTEXT EXEC

ZAPTEXT EXEC modifies or dumps individual text files. Use ZAPTEXT like the ZAP service program, but only for text files, not for MODULEs, TXTLIBs, or LOADLIBs. (Use ZAP to process MODULEs, TXTLIBs, and LOADLIBs.) ZAPTEXT uses the same control information as ZAP and can also use the EXPAND ZAP control word. The user's A-disk must be accessed as read/write to use ZAPTEXT.

Format

The format of the ZAPTEXT EXEC is:

ZAPTEXT	*fn [ft [fm]] [(options: ... [)]]* *options:* [INPUT *filename*] [PRINT] [NOPRINT]

Where:

fn ft fm
is the file ID of the text file that you want to change. If you do not specify the filetype or filemode, the system assumes a filetype of TEXT and a filemode of A1. The filemode must specify a read/write disk.

options:

INPUT *filename*
identifies the file that has the ZAP control records. This file must:

- Have a filetype of ZAP, and
- Be a fixed 80-byte sequential file that resides on any accessible disk.

If you do not specify *filename*, it defaults to whatever you specify as *fn* on the ZAPTEXT command.

PRINT
prints all output produced by ZAPTEXT on the printer. The system also displays error messages, commands in error, and control records in error at the terminal.

NOPRINT
does not print anything on the printer, and instead displays all output at the terminal.

ZAPTEXT Input Control Records

Control Records Also Used by ZAP

ZAPTEXT uses the same control information as the current ZAP service program, with the addition of the EXPAND control record. The ZAP service program ignores any EXPAND control records. Refer to "Input Control Records" on page 713 for information about the control records, other than EXPAND, that ZAPTEXT uses.

Use the ZAP control records with ZAPTEXT according to ZAP's TXTLIB conventions.

EXPAND Control Record

The EXPAND control record lets you increase the size of a named control section contained in the text file.

Format:

```
EXPAND          csect size [ , csect size ...]
```

Where:

csect
specifies the symbolic name of a control section whose length you want to increase.

size
specifies the decimal number of bytes for the system to add to the control section length. The system initializes the added bytes to binary zero. The maximum number of bytes for each control section that you indicate is 4095.

Control Record Usage Notes

1. Each control record may have multiple entries, but you must separate them with commas. Do not spill an entry onto the next line.
2. The system processes all EXPAND control records before any other control records, regardless of their position in the control file.
3. The effective length of the expansion, which is the actual number of bytes added to the control section, may be greater than the length that you specify for the expansion. This may occur if, after the specified expansion, the system must add padding bytes to align the next control section or common area.
4. When you increase a control section's size, it may affect the offset address of any following control section. This is important when you determine values for BASE, REP, and VER control records. Use the effective expansion lengths when you are determining control section offsets.

EXPAND Command

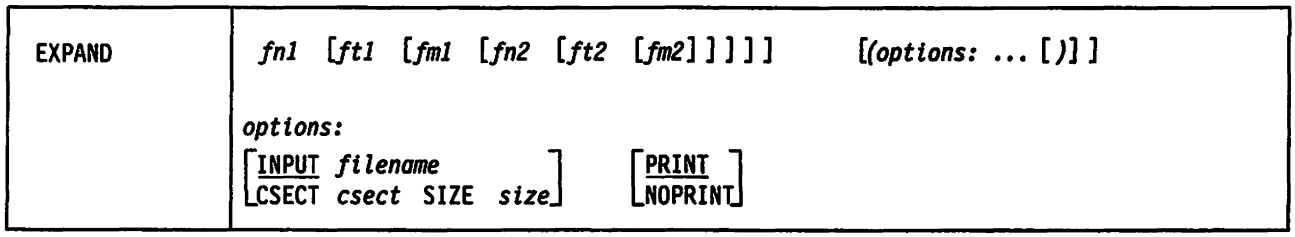
ZAPTEXT calls EXPAND if you specify an EXPAND control record in the ZAP control file. Use EXPAND to add space to a program in object deck form. The system creates object decks when you assemble or compile a source program. This is especially useful when you do not have the source code for a program or the program does not have a patch area.

Note: EXPAND can add extra space only at the end of named control sections (CSECTs). EXPAND cannot expand private code (unnamed CSECT) and common areas (named or unnamed).

Do not increase the length of a program beyond its design limitations. For example, if you add space to a control section beyond the range of its base register addressability, that space is unusable.

Format

The format of the EXPAND command is:



Where:

fn1 ft1 fm1
identifies the input text file that the system expands. The file must have valid object deck information, like that created by an assembler or compiler. If you do not specify the filetype or filemode, the system assumes a filetype of TEXT and filemode of A1.

Note: EXPAND assumes that the input text file follows OS/VS standards and that the OS/VS Linkage Editor will accept it without error. The system does a limited check for errors. If the input file is invalid, the system may not expand the text file correctly.

fn2 ft2 fm2
identifies the output text file that the system creates. You can use an equal sign (=) for any of the file identifiers to indicate that it is the same identifier as *fn1*, *ft1*, or *fm1*. The default fileid is *$fn1* = =. The system truncates the filename (*fn1*) to 7 characters before appending the $. In any case, *fm2* must be a read/write disk and cannot be an asterisk (*).

***options*:**

INPUT *filename*
identifies an EXPAND input file that contains EXPAND control records. If you do not specify INPUT, the filename defaults to the name of the text file that you are expanding (*fn1*). The filetype must be EXPAND. The system searches all accessed disks for this file.

Do not specify this option with the CSECT or SIZE options.

CSECT
specifies the symbolic name of a control section whose length the system will increase. If you specify CSECT, you must also specify SIZE.

Do not specify CSECT with the INPUT option.

SIZE
specifies the decimal number of bytes that the system adds to the control section length. The system initializes the added bytes to binary zeros. The maximum number of bytes for each control section is 4095. If you specify SIZE, you must also specify CSECT.

Do not specify SIZE with the INPUT option.

PRINT
prints on the printer all output that EXPAND produces. In addition, the system displays error messages, commands in error, and control records in error at the terminal.

NOPRINT
does not print any output on the printer, and instead displays it at the terminal.

Messages

- After the system expands each CSECT that you specified, the system issues a message indicating the following:
  - The number of bytes added to the control section.
  - Whether the number of bytes added is greater than the length that you specify for the expansion. This may occur if, after the specified expansion, the system must add padding bytes to align the next control section or common area.
  - The offset, relative to the start of the specified control section where the expansion began.
- If the system finds an error during processing, it stops the update and does not do the expansions.

Appendix C. VM/XA System Product Starter System Information

This section presents the following information:

- The minimum hardware configuration needed to generate VM/XA System Product.
- Sample files that IBM provides with the VM/XA System Product. (For a sample product parameter file, see "A Sample Product Parameter File" on page 384.).
- Allocations of the system residence device for starter systems.
- Minidisk maps of 3350, 3375, 3380, 3380-E4, 3380-K, 3390, and 3391 system residence devices, based on sample directories.

Minimum Hardware Configuration

The minimum hardware configuration needed to generate VM/XA System Product is:

- One processor (that supports VM/XA SP) with:
  - Processor in 370-XA mode
  - At least 4 megabytes of real storage
  - One processor unit
  - One processor controller
  - One channel processor
- One system console.
- One printer.
- Direct access storage devices. The number required depends on the type of DASD used.

DASD Type	Number of DASD
3350	4
3375	3
3380	2
3380-E4	1
3380-K	1
3390	1

- One magnetic 9-track tape unit with data density of 6250 BPI. (A second tape unit is recommended, but not required.)
- One operator's console.

IBM supplies the following VM/XA SP starter systems:

3350
3375
3380 (for 3380, 3380-E4, and 3380-K DASD)
3390

DMSNGP ASSEMBLE (CMS Nucleus Generation Profile)

This profile is part of the Starter System. Within this file, you must change CYLADDR=? to the address of the starting cylinder or block where the nucleus is to be written. Also, you may want to change the VERSION and INSTID names to whatever is appropriate for your installation.

```
NGP      TITLE 'DMSNGP     (CMS)      VM/XA SYSTEM PRODUCT 5664-308'
         EJECT
DMSNGP   CSECT
         DEFNUC SYSDISK=190,        * S-disk address                 *
               YDISK=19E,           * Y-disk address                 *
               HELP=19D,            * Help disk address              *
               LANGID=AMENG,        * Default is American English    *
               DBCS=NO,             * Default is not a DBCS lang     *
               LANGLEV=S,           * DCSS ID for multiple DCSS      *
               SAVESYS=NO,          * Using CMS in DCSS YES OR NO    *
               SYSNAME=CMS,         * Name of above DCSS to save     *
               USEINST=YES,         * Using EXEC/XEDIT in DCSS       *
               INSTSEG=CMSINST,     * Name of above DCSS to save     *
               REWRITE=YES,         * Write nucleus yes or no        *
               IPLADDR=190,         * Address of where to write      *
               CYLADDR=?,           * CYL/BLK of where to write      *
               IPLCYL0=YES,         * write ipl text on cyl 0        *
               VERSION=?,           * VM/XA CMS 5.6 MM/DD/YY HH MM SS *
               INSTID='VM/XA CONVERSATIONAL MONITOR SYSTEM'
         END
```

Sample HCPRIO ASSEMBLE File

IBM supplies the following sample file with VM/XA SP. **It is only a sample file, and must be tailored to your installation before it can be used.**

```
RIO TITLE 'HCPRIOXA - HCPRIO FOR VM/XA SYSTEM PRODUCT   '
*****************************************************************
***     Virtual Machine / System Product       5664-308    ***
***     Copyright (c) I B M   Corporation      1988, 1989  ***
***     Licensed Materials -  Property of I B M            ***
***     Refer to Copyright Instructions: Form G120-2083    ***
*****************************************************************
*****************************************************************
*
* MODULE NAME - HCPRIO
*
* DESCRIPTIVE NAME - DEFINITION OF REAL DEVICES
*
*
* FUNCTION - TO DEFINE THE REAL DEVICES
*
*   REGISTER CONVENTIONS - SYMBOLIC REFERENCES TO REGISTERS ARE
*                          OF THE FORM "RX" WHERE X IS A NUMBER
*                          RANGING FROM 0 TO 15.
*
*   PATCH LABEL - NONE
*
* MODULE TYPE - DATA
*
*   PROCESSOR - ASSEMBLER H (VERSION 2)
*
*   ATTRIBUTES - RESIDENT, DATA-ONLY
*
* MACROS -
*
*          RDEVICE  - DEFINE REAL DEVICES
*          RIOGEN   - END REAL DEVICE GENERATION
*
*
* GENERAL COMMENTS - THIS MODULE IS THE USER'S RESPONSIBILITY.
*                    THIS SAMPLE IS BASED UPON THE STARTER IOCP
*                    INPUT FILE.
*
*********************************************************************
         PRINT NOGEN
         EJECT
*-------------------------------------------------------------
* THE RDEVICE MACROS ARE CODED HERE, ONE FOR EACH DEVICE OR
* GROUP OF DEVICES
*-------------------------------------------------------------
*
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(070,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(170,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(270,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(370,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(470,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(570,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(670,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(770,8)
```

```
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(870,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(970,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(A70,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(B70,8)
*
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(080,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(180,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(280,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(380,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(480,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(580,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(680,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(780,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(880,8)
 RDEVICE DEVTYPE=3420,FEATURE=DUALDENS,MODEL=6,DEVNO=(980,8)
*
 RDEVICE DEVTYPE=3422,DEVNO=(A80,8)
 RDEVICE DEVTYPE=3422,DEVNO=(B80,8)
*
 RDEVICE DEVTYPE=3430,DEVNO=(C80,8)
 RDEVICE DEVTYPE=3430,DEVNO=(D80,8)
*
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(078,8)
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(178,8)
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(278,8)
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(378,8)
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(478,8)
 RDEVICE DEVTYPE=3330,MODEL=1,DEVNO=(578,8)
*
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(050,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(150,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(250,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(350,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(450,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(550,8)
*
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(058,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(158,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(258,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(358,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(458,8)
 RDEVICE DEVTYPE=3330,MODEL=11,DEVNO=(558,8)
*
 RDEVICE DEVTYPE=3340,DEVNO=(0C0,8)
 RDEVICE DEVTYPE=3340,DEVNO=(1C0,8)
 RDEVICE DEVTYPE=3340,DEVNO=(2C0,8)
 RDEVICE DEVTYPE=3340,DEVNO=(3C0,8)
 RDEVICE DEVTYPE=3340,DEVNO=(4C0,8)
 RDEVICE DEVTYPE=3340,DEVNO=(5C0,8)
*
 RDEVICE DEVTYPE=3350,DEVNO=(048,8)
 RDEVICE DEVTYPE=3350,DEVNO=(148,8)
 RDEVICE DEVTYPE=3350,DEVNO=(248,8)
 RDEVICE DEVTYPE=3350,DEVNO=(348,8)
 RDEVICE DEVTYPE=3350,DEVNO=(448,8)
 RDEVICE DEVTYPE=3350,DEVNO=(548,8)
*
 RDEVICE DEVTYPE=3350,DEVNO=(030,8)
 RDEVICE DEVTYPE=3350,DEVNO=(130,8)
 RDEVICE DEVTYPE=3350,DEVNO=(230,8)
```

```
 RDEVICE DEVTYPE=3350,DEVNO=(330,8)
 RDEVICE DEVTYPE=3350,DEVNO=(430,8)
 RDEVICE DEVTYPE=3350,DEVNO=(530,8)
*
 RDEVICE DEVTYPE=3350,DEVNO=(060,8)
 RDEVICE DEVTYPE=3350,DEVNO=(160,8)
 RDEVICE DEVTYPE=3350,DEVNO=(260,8)
 RDEVICE DEVTYPE=3350,DEVNO=(360,8)
 RDEVICE DEVTYPE=3350,DEVNO=(460,8)
 RDEVICE DEVTYPE=3350,DEVNO=(560,8)
*
 RDEVICE DEVTYPE=3375,DEVNO=(090,8)
 RDEVICE DEVTYPE=3375,DEVNO=(190,8)
 RDEVICE DEVTYPE=3375,DEVNO=(290,8)
 RDEVICE DEVTYPE=3375,DEVNO=(390,8)
 RDEVICE DEVTYPE=3375,DEVNO=(490,8)
*
 RDEVICE DEVTYPE=3375,DEVNO=(068,8)
 RDEVICE DEVTYPE=3375,DEVNO=(168,8)
 RDEVICE DEVTYPE=3375,DEVNO=(268,8)
 RDEVICE DEVTYPE=3375,DEVNO=(368,8)
 RDEVICE DEVTYPE=3375,DEVNO=(468,8)
 RDEVICE DEVTYPE=3375,DEVNO=(568,8)
 RDEVICE DEVTYPE=3375,DEVNO=(668,8)
*
 RDEVICE DEVTYPE=3380,DEVNO=(038,8)
 RDEVICE DEVTYPE=3380,DEVNO=(138,8)
 RDEVICE DEVTYPE=3380,DEVNO=(238,8)
 RDEVICE DEVTYPE=3380,DEVNO=(338,8)
 RDEVICE DEVTYPE=3380,DEVNO=(438,8)
 RDEVICE DEVTYPE=3380,DEVNO=(538,8)
 RDEVICE DEVTYPE=3380,DEVNO=(638,8)
*
 RDEVICE DEVTYPE=3380,DEVNO=(0A0,8)
 RDEVICE DEVTYPE=3380,DEVNO=(1A0,8)
 RDEVICE DEVTYPE=3380,DEVNO=(2A0,8)
 RDEVICE DEVTYPE=3380,DEVNO=(3A0,8)
 RDEVICE DEVTYPE=3380,DEVNO=(4A0,8)
*
 RDEVICE DEVTYPE=3390,DEVNO=(5A0,8)
 RDEVICE DEVTYPE=3390,DEVNO=(6A0,8)
 RDEVICE DEVTYPE=3390,DEVNO=(7A0,8)
*
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(0F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(1F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(2F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(3F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(4F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(5F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(6F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(7F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(8F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(9F0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(AF0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(BF0,1)
*
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(0D0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(1D0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(2D0,1)
 RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(3D0,1)
```

```
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(4D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(5D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(6D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(7D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(8D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(9D0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(AD0,1)
|   RDEVICE DEVTYPE=2305,MODEL=2,DEVNO=(BD0,1)
|  *
|   RDEVICE DEVTYPE=3800,DEVNO=(018,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(118,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(218,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(318,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(418,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(518,1)
|   RDEVICE DEVTYPE=3800,DEVNO=(618,1)
|  *
|   RDEVICE DEVTYPE=3262,DEVNO=(0A8,1)
|   RDEVICE DEVTYPE=3262,DEVNO=(1A8,1)
|   RDEVICE DEVTYPE=3262,DEVNO=(2A8,1)
|   RDEVICE DEVTYPE=3262,DEVNO=(3A8,1)
|   RDEVICE DEVTYPE=3262,DEVNO=(4A8,1)
|   RDEVICE DEVTYPE=3262,DEVNO=(5A8,1)
|  *
|   RDEVICE DEVTYPE=4245,DEVNO=(0B0,1)
|   RDEVICE DEVTYPE=4245,DEVNO=(1B0,1)
|   RDEVICE DEVTYPE=4245,DEVNO=(2B0,1)
|   RDEVICE DEVTYPE=4245,DEVNO=(3B0,1)
|   RDEVICE DEVTYPE=4245,DEVNO=(4B0,1)
|   RDEVICE DEVTYPE=4245,DEVNO=(5B0,1)
|  *
|   RDEVICE DEVTYPE=4248,DEVNO=(0B8,1)
|   RDEVICE DEVTYPE=4248,DEVNO=(1B8,1)
|   RDEVICE DEVTYPE=4248,DEVNO=(2B8,1)
|   RDEVICE DEVTYPE=4248,DEVNO=(3B8,1)
|   RDEVICE DEVTYPE=4248,DEVNO=(4B8,1)
|   RDEVICE DEVTYPE=4248,DEVNO=(5B8,1)
|  *
|   RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(020,8)
|   RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(120,8)
|   RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(220,8)
|   RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(320,8)
|   RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(420,8)
|  *
|   RDEVICE DEVTYPE=1403,DEVNO=(00E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(10E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(20E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(30E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(40E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(50E,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(60E,1)
|  *
|   RDEVICE DEVTYPE=1403,DEVNO=(00F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(10F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(20F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(30F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(40F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(50F,1)
|   RDEVICE DEVTYPE=1403,DEVNO=(60F,1)
|  *
```

```
  RDEVICE DEVTYPE=2540R,DEVNO=(00C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(10C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(20C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(30C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(40C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(50C,1)
  RDEVICE DEVTYPE=2540R,DEVNO=(60C,1)
 *
  RDEVICE DEVTYPE=2540P,DEVNO=(00D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(10D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(20D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(30D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(40D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(50D,1)
  RDEVICE DEVTYPE=2540P,DEVNO=(60D,1)
 *
  RDEVICE DEVTYPE=3211,DEVNO=(002,1)
  RDEVICE DEVTYPE=3211,DEVNO=(102,1)
  RDEVICE DEVTYPE=3211,DEVNO=(202,1)
  RDEVICE DEVTYPE=3211,DEVNO=(302,1)
  RDEVICE DEVTYPE=3211,DEVNO=(402,1)
 *
  RDEVICE DEVTYPE=3203,MODEL=5,DEVNO=(003,1)
  RDEVICE DEVTYPE=3203,MODEL=5,DEVNO=(103,1)
  RDEVICE DEVTYPE=3203,MODEL=5,DEVNO=(203,1)
  RDEVICE DEVTYPE=3203,MODEL=5,DEVNO=(303,1)
 *
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(040,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(140,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(240,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(340,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(440,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(540,8)
  RDEVICE DEVTYPE=3278,MODEL=2,DEVNO=(640,8)
 *
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(088,8)
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(188,8)
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(288,8)
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(388,8)
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(488,8)
  RDEVICE DEVTYPE=3278,MODEL=5,DEVNO=(588,8)
 *
  RDEVICE DEVTYPE=3290,DEVNO=(098,8)
  RDEVICE DEVTYPE=3290,DEVNO=(198,8)
  RDEVICE DEVTYPE=3290,DEVNO=(298,8)
  RDEVICE DEVTYPE=3290,DEVNO=(398,8)
  RDEVICE DEVTYPE=3290,DEVNO=(498,8)
  RDEVICE DEVTYPE=3290,DEVNO=(598,8)
 *
  RDEVICE DEVTYPE=3505,CLASS=A,DEVNO=(012,1)
  RDEVICE DEVTYPE=3505,CLASS=A,DEVNO=(112,1)
  RDEVICE DEVTYPE=3505,CLASS=A,DEVNO=(212,1)
  RDEVICE DEVTYPE=3505,CLASS=A,DEVNO=(312,1)
  RDEVICE DEVTYPE=3505,CLASS=A,DEVNO=(412,1)
 *
  RDEVICE DEVTYPE=3525,CLASS=A,DEVNO=(013,1)
  RDEVICE DEVTYPE=3525,CLASS=A,DEVNO=(113,1)
  RDEVICE DEVTYPE=3525,CLASS=A,DEVNO=(213,1)
  RDEVICE DEVTYPE=3525,CLASS=A,DEVNO=(313,1)
```

```
 RDEVICE DEVTYPE=3525,CLASS=A,DEVNO=(413,1)
*
 RDEVICE DEVTYPE=3480,DEVNO=(004,4)
 RDEVICE DEVTYPE=3480,DEVNO=(104,4)
 RDEVICE DEVTYPE=3480,DEVNO=(204,4)
 RDEVICE DEVTYPE=3480,DEVNO=(304,4)
*
 RDEVICE DEVTYPE=3890,DEVNO=(008,4)
 RDEVICE DEVTYPE=3890,DEVNO=(108,4)
*
 RDEVICE DEVTYPE=HFGD,DEVNO=(014,4)
 RDEVICE DEVTYPE=HFGD,DEVNO=(114,4)
*
  EJECT
*****************************************************************
*---------------------------------------------------------------
* THE RIOGEN MACRO IS CODED HERE
*---------------------------------------------------------------
         SPACE 2
         RIOGEN CONS=020,ALTCONS=(120,220,320,                  *
               420,040,140,240,340,440,540,640)
```

Sample Input/Output Configuration Profile

IBM supplies the following sample file with VM/XA SP. **It is only a sample file, and must be tailored to your installation before it can be used.**

```
************************************************************
***     Virtual Machine / System Product       5664-308    ***
***     Copyright (c) I B M   Corporation      1988, 1989  ***
***     Licensed Materials -  Property of I B M            ***
***     Refer to Copyright Instructions: Form G120-2083    ***
************************************************************
 ID MSG1='SAMPLE IOCP INPUT FILE',                                    *
               MSG2='COMPANION TO SAMPLE HCPRIO'
 CHPID PATH=((00,0,0),(01,1,0),(02,2,0),(03,3,0),(04,4,0),(05,5,0),   *
               (06,6,0),(07,7,0)),TYPE=BL
 CHPID PATH=((10,8,0),(11,9,0),(12,A,0),(13,B,0)),TYPE=BL
 CHPID PATH=((14,0,1),(15,1,1),(16,2,1),(17,3,1),(20,4,1),(21,5,1),   *
               (22,6,1),(23,7,1)),TYPE=BL
 CHPID PATH=((24,8,1),(25,9,1),(26,A,1),(27,B,1)),TYPE=BL
*IOCP
 CNTLUNIT CUNUMBR=001,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(00,14),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=002,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(01,15),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=003,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(02,16),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=004,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(03,17),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=005,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(04,20),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=006,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(05,21),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=007,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(06,22),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=008,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(07,23),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=009,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(10,24),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=00A,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(11,25),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=00B,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(12,26),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=00C,UNIT=3803,UNITADD=((70,8)),                     *
               PATH=(13,27),PROTOCL=D,SHARED=Y
*IOCP
 CNTLUNIT CUNUMBR=00D,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(00,14),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=00E,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(01,15),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=00F,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(02,16),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=010,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(03,17),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=011,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(04,20),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=012,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(05,21),PROTOCL=D,SHARED=Y
 CNTLUNIT CUNUMBR=013,UNIT=3803,UNITADD=((80,8)),                     *
               PATH=(06,22),PROTOCL=D,SHARED=Y
```

```
  CNTLUNIT CUNUMBR=014,UNIT=3803,UNITADD=((80,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=015,UNIT=3803,UNITADD=((80,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=016,UNIT=3803,UNITADD=((80,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=017,UNIT=3803,UNITADD=((80,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=018,UNIT=3803,UNITADD=((80,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=Y
*
  CNTLUNIT CUNUMBR=019,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01A,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01B,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01C,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01D,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01E,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=01F,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=020,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=021,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=022,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=023,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=024,UNIT=3830,UNITADD=((50,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=025,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=026,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=027,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=028,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=029,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02A,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02B,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02C,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02D,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02E,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=02F,UNIT=3830,UNITADD=((58,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=030,UNIT=3830,UNITADD=((58,8)),                *
```

```
|               PATH=(13,27),PROTOCL=D,SHARED=N
| *
|  CNTLUNIT CUNUMBR=031,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(00,14),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=032,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(01,15),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=033,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(02,16),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=034,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(03,17),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=035,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(04,20),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=036,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(05,21),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=037,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(06,22),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=038,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(07,23),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=039,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(10,24),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=03A,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(11,25),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=03B,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(12,26),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=03C,UNIT=3830,UNITADD=((C0,8)),                    *
|               PATH=(13,27),PROTOCL=D,SHARED=N
| *
|  CNTLUNIT CUNUMBR=03D,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(00,14),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=03E,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(01,15),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=03F,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(02,16),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=040,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(03,17),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=041,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(04,20),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=042,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(05,21),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=043,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(06,22),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=044,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(07,23),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=045,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(10,24),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=046,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(11,25),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=047,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(12,26),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=048,UNIT=3830,UNITADD=((48,8)),                    *
|               PATH=(13,27),PROTOCL=D,SHARED=N
| *
|  CNTLUNIT CUNUMBR=049,UNIT=3830,UNITADD=((30,8)),                    *
|               PATH=(00,14),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=04A,UNIT=3830,UNITADD=((30,8)),                    *
|               PATH=(01,15),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=04B,UNIT=3830,UNITADD=((30,8)),                    *
|               PATH=(02,16),PROTOCL=D,SHARED=N
|  CNTLUNIT CUNUMBR=04C,UNIT=3830,UNITADD=((30,8)),                    *
|               PATH=(03,17),PROTOCL=D,SHARED=N
```

```
  CNTLUNIT CUNUMBR=04D,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=04E,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=04F,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=050,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=051,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=052,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=053,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=054,UNIT=3830,UNITADD=((30,8)),                    *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=055,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=056,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=057,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=058,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=059,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05A,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05B,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05C,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05D,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05E,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=05F,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=060,UNIT=3880,UNITADD=((60,8)),                    *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=061,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=062,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=063,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=064,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=065,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=066,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=067,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=068,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=069,UNIT=2835,UNITADD=((F0,8)),                    *
                PATH=(10,24),PROTOCL=D,SHARED=N
```

```
  CNTLUNIT CUNUMBR=06A,UNIT=2835,UNITADD=((F0,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=06B,UNIT=2835,UNITADD=((F0,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=06C,UNIT=2835,UNITADD=((F0,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=06D,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=06E,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=06F,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=070,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=071,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=072,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=073,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=074,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=075,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=076,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=077,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=078,UNIT=2835,UNITADD=((D0,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=079,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07A,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07B,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07C,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07D,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07E,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=07F,UNIT=3800,UNITADD=((18,1)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=080,UNIT=3274,UNITADD=((20,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=081,UNIT=3274,UNITADD=((20,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=082,UNIT=3274,UNITADD=((20,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=083,UNIT=3274,UNITADD=((20,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=084,UNIT=3274,UNITADD=((20,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=Y
*
  CNTLUNIT CUNUMBR=085,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
```

```
  CNTLUNIT CUNUMBR=086,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=087,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=088,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=089,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=08A,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=08B,UNIT=2821,UNITADD=((0C,4)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
 *
  CNTLUNIT CUNUMBR=08C,UNIT=3811,UNITADD=((02,1)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=08D,UNIT=3811,UNITADD=((02,1)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=08E,UNIT=3811,UNITADD=((02,1)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=08F,UNIT=3811,UNITADD=((02,1)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=090,UNIT=3811,UNITADD=((02,1)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
 *
  CNTLUNIT CUNUMBR=091,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=092,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=093,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=094,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=095,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=096,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=Y
  CNTLUNIT CUNUMBR=097,UNIT=3274,UNITADD=((40,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=Y
 *
  CNTLUNIT CUNUMBR=098,UNIT=3505,UNITADD=((12,2)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=099,UNIT=3505,UNITADD=((12,2)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=09A,UNIT=3505,UNITADD=((12,2)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=09B,UNIT=3505,UNITADD=((12,2)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=09C,UNIT=3505,UNITADD=((12,2)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
 *
  CNTLUNIT CUNUMBR=09D,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=09E,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=09F,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A0,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A1,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
```

```
  CNTLUNIT CUNUMBR=0A2,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A3,UNIT=3880,UNITADD=((68,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0A4,UNIT=3880,UNITADD=((90,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A5,UNIT=3880,UNITADD=((90,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A6,UNIT=3880,UNITADD=((90,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A7,UNIT=3880,UNITADD=((90,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0A8,UNIT=3880,UNITADD=((90,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0A9,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AA,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AB,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AC,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AD,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(04,20),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AE,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(05,21),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0AF,UNIT=3880,UNITADD=((38,8)),                *
                PATH=(06,22),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0B0,UNIT=3880,UNITADD=((A0,8)),                *
                PATH=(07,23),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B1,UNIT=3880,UNITADD=((A0,8)),                *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B2,UNIT=3880,UNITADD=((A0,8)),                *
                PATH=(11,25),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B3,UNIT=3880,UNITADD=((A0,8)),                *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B4,UNIT=3880,UNITADD=((A0,8)),                *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0B5,UNIT=3203,UNITADD=((03,1)),                *
                PATH=(00,14),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B6,UNIT=3203,UNITADD=((03,1)),                *
                PATH=(01,15),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B7,UNIT=3203,UNITADD=((03,1)),                *
                PATH=(02,16),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0B8,UNIT=3203,UNITADD=((03,1)),                *
                PATH=(03,17),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0B9,UNIT=3480,UNITADD=((04,4)),                *
                PATH=(04,20),PROTOCL=S,SHARED=N
  CNTLUNIT CUNUMBR=0BA,UNIT=3480,UNITADD=((04,4)),                *
                PATH=(05,21),PROTOCL=S,SHARED=N
  CNTLUNIT CUNUMBR=0BB,UNIT=3480,UNITADD=((04,4)),                *
                PATH=(06,22),PROTOCL=S,SHARED=N
  CNTLUNIT CUNUMBR=0BC,UNIT=3480,UNITADD=((04,4)),                *
                PATH=(07,23),PROTOCL=S,SHARED=N
*
```

```
  CNTLUNIT CUNUMBR=0BD,UNIT=3890,UNITADD=((08,4)),                    *
                PATH=(10,24),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0BE,UNIT=3890,UNITADD=((08,4)),                    *
                PATH=(11,25),PROTOCL=D,SHARED=N
*
  CNTLUNIT CUNUMBR=0BF,UNIT=5088,UNITADD=((14,4)),                    *
                PATH=(12,26),PROTOCL=D,SHARED=N
  CNTLUNIT CUNUMBR=0C0,UNIT=5088,UNITADD=((14,4)),                    *
                PATH=(13,27),PROTOCL=D,SHARED=N
*
  IODEVICE CUNUMBR=001,UNIT=3420,ADDRESS=(070,8)
  IODEVICE CUNUMBR=002,UNIT=3420,ADDRESS=(170,8)
  IODEVICE CUNUMBR=003,UNIT=3420,ADDRESS=(270,8)
  IODEVICE CUNUMBR=004,UNIT=3420,ADDRESS=(370,8)
  IODEVICE CUNUMBR=005,UNIT=3420,ADDRESS=(470,8)
  IODEVICE CUNUMBR=006,UNIT=3420,ADDRESS=(570,8)
  IODEVICE CUNUMBR=007,UNIT=3420,ADDRESS=(670,8)
  IODEVICE CUNUMBR=008,UNIT=3420,ADDRESS=(770,8)
  IODEVICE CUNUMBR=009,UNIT=3420,ADDRESS=(870,8)
  IODEVICE CUNUMBR=00A,UNIT=3420,ADDRESS=(970,8)
  IODEVICE CUNUMBR=00B,UNIT=3420,ADDRESS=(A70,8)
  IODEVICE CUNUMBR=00C,UNIT=3420,ADDRESS=(B70,8)
*
  IODEVICE CUNUMBR=00D,UNIT=3420,ADDRESS=(080,8)
  IODEVICE CUNUMBR=00E,UNIT=3420,ADDRESS=(180,8)
  IODEVICE CUNUMBR=00F,UNIT=3420,ADDRESS=(280,8)
  IODEVICE CUNUMBR=010,UNIT=3420,ADDRESS=(380,8)
  IODEVICE CUNUMBR=011,UNIT=3420,ADDRESS=(480,8)
  IODEVICE CUNUMBR=012,UNIT=3420,ADDRESS=(580,8)
  IODEVICE CUNUMBR=013,UNIT=3420,ADDRESS=(680,8)
  IODEVICE CUNUMBR=014,UNIT=3420,ADDRESS=(780,8)
  IODEVICE CUNUMBR=015,UNIT=3420,ADDRESS=(880,8)
  IODEVICE CUNUMBR=016,UNIT=3420,ADDRESS=(980,8)
  IODEVICE CUNUMBR=017,UNIT=3420,ADDRESS=(A80,8)
  IODEVICE CUNUMBR=018,UNIT=3420,ADDRESS=(B80,8)
*
  IODEVICE CUNUMBR=019,UNIT=3330,ADDRESS=(050,8)
  IODEVICE CUNUMBR=01A,UNIT=3330,ADDRESS=(150,8)
  IODEVICE CUNUMBR=01B,UNIT=3330,ADDRESS=(250,8)
  IODEVICE CUNUMBR=01C,UNIT=3330,ADDRESS=(350,8)
  IODEVICE CUNUMBR=01D,UNIT=3330,ADDRESS=(450,8)
  IODEVICE CUNUMBR=01E,UNIT=3330,ADDRESS=(550,8)
  IODEVICE CUNUMBR=01F,UNIT=3330,ADDRESS=(650,8)
  IODEVICE CUNUMBR=020,UNIT=3330,ADDRESS=(750,8)
  IODEVICE CUNUMBR=021,UNIT=3330,ADDRESS=(850,8)
  IODEVICE CUNUMBR=022,UNIT=3330,ADDRESS=(950,8)
  IODEVICE CUNUMBR=023,UNIT=3330,ADDRESS=(A50,8)
  IODEVICE CUNUMBR=024,UNIT=3330,ADDRESS=(B50,8)
*
  IODEVICE CUNUMBR=025,UNIT=3330,ADDRESS=(058,8)
  IODEVICE CUNUMBR=026,UNIT=3330,ADDRESS=(158,8)
  IODEVICE CUNUMBR=027,UNIT=3330,ADDRESS=(258,8)
  IODEVICE CUNUMBR=028,UNIT=3330,ADDRESS=(358,8)
  IODEVICE CUNUMBR=029,UNIT=3330,ADDRESS=(458,8)
  IODEVICE CUNUMBR=02A,UNIT=3330,ADDRESS=(558,8)
  IODEVICE CUNUMBR=02B,UNIT=3330,ADDRESS=(658,8)
  IODEVICE CUNUMBR=02C,UNIT=3330,ADDRESS=(758,8)
  IODEVICE CUNUMBR=02D,UNIT=3330,ADDRESS=(858,8)
  IODEVICE CUNUMBR=02E,UNIT=3330,ADDRESS=(958,8)
  IODEVICE CUNUMBR=02F,UNIT=3330,ADDRESS=(A58,8)
```

```
  IODEVICE CUNUMBR=030,UNIT=3330,ADDRESS=(B58,8)
 *
  IODEVICE CUNUMBR=031,UNIT=3340,ADDRESS=(0C0,8)
  IODEVICE CUNUMBR=032,UNIT=3340,ADDRESS=(1C0,8)
  IODEVICE CUNUMBR=033,UNIT=3340,ADDRESS=(2C0,8)
  IODEVICE CUNUMBR=034,UNIT=3340,ADDRESS=(3C0,8)
  IODEVICE CUNUMBR=035,UNIT=3340,ADDRESS=(4C0,8)
  IODEVICE CUNUMBR=036,UNIT=3340,ADDRESS=(5C0,8)
  IODEVICE CUNUMBR=037,UNIT=3340,ADDRESS=(6C0,8)
  IODEVICE CUNUMBR=038,UNIT=3340,ADDRESS=(7C0,8)
  IODEVICE CUNUMBR=039,UNIT=3340,ADDRESS=(8C0,8)
  IODEVICE CUNUMBR=03A,UNIT=3340,ADDRESS=(9C0,8)
  IODEVICE CUNUMBR=03B,UNIT=3340,ADDRESS=(AC0,8)
  IODEVICE CUNUMBR=03C,UNIT=3340,ADDRESS=(BC0,8)
 *
  IODEVICE CUNUMBR=03D,UNIT=3350,ADDRESS=(048,8)
  IODEVICE CUNUMBR=03E,UNIT=3350,ADDRESS=(148,8)
  IODEVICE CUNUMBR=03F,UNIT=3350,ADDRESS=(248,8)
  IODEVICE CUNUMBR=040,UNIT=3350,ADDRESS=(348,8)
  IODEVICE CUNUMBR=041,UNIT=3350,ADDRESS=(448,8)
  IODEVICE CUNUMBR=042,UNIT=3350,ADDRESS=(548,8)
  IODEVICE CUNUMBR=043,UNIT=3350,ADDRESS=(648,8)
  IODEVICE CUNUMBR=044,UNIT=3350,ADDRESS=(748,8)
  IODEVICE CUNUMBR=045,UNIT=3350,ADDRESS=(848,8)
  IODEVICE CUNUMBR=046,UNIT=3350,ADDRESS=(948,8)
  IODEVICE CUNUMBR=047,UNIT=3350,ADDRESS=(A48,8)
  IODEVICE CUNUMBR=048,UNIT=3350,ADDRESS=(B48,8)
 *
  IODEVICE CUNUMBR=049,UNIT=3350,ADDRESS=(030,8)
  IODEVICE CUNUMBR=04A,UNIT=3350,ADDRESS=(130,8)
  IODEVICE CUNUMBR=04B,UNIT=3350,ADDRESS=(230,8)
  IODEVICE CUNUMBR=04C,UNIT=3350,ADDRESS=(330,8)
  IODEVICE CUNUMBR=04D,UNIT=3350,ADDRESS=(430,8)
  IODEVICE CUNUMBR=04E,UNIT=3350,ADDRESS=(530,8)
  IODEVICE CUNUMBR=04F,UNIT=3350,ADDRESS=(630,8)
  IODEVICE CUNUMBR=050,UNIT=3350,ADDRESS=(730,8)
  IODEVICE CUNUMBR=051,UNIT=3350,ADDRESS=(830,8)
  IODEVICE CUNUMBR=052,UNIT=3350,ADDRESS=(930,8)
  IODEVICE CUNUMBR=053,UNIT=3350,ADDRESS=(A30,8)
  IODEVICE CUNUMBR=054,UNIT=3350,ADDRESS=(B30,8)
 *
  IODEVICE CUNUMBR=055,UNIT=3350,ADDRESS=(060,8)
  IODEVICE CUNUMBR=056,UNIT=3350,ADDRESS=(160,8)
  IODEVICE CUNUMBR=057,UNIT=3350,ADDRESS=(260,8)
  IODEVICE CUNUMBR=058,UNIT=3350,ADDRESS=(360,8)
  IODEVICE CUNUMBR=059,UNIT=3350,ADDRESS=(460,8)
  IODEVICE CUNUMBR=05A,UNIT=3350,ADDRESS=(560,8)
  IODEVICE CUNUMBR=05B,UNIT=3350,ADDRESS=(660,8)
  IODEVICE CUNUMBR=05C,UNIT=3350,ADDRESS=(760,8)
  IODEVICE CUNUMBR=05D,UNIT=3350,ADDRESS=(860,8)
  IODEVICE CUNUMBR=05E,UNIT=3350,ADDRESS=(960,8)
  IODEVICE CUNUMBR=05F,UNIT=3350,ADDRESS=(A60,8)
  IODEVICE CUNUMBR=060,UNIT=3350,ADDRESS=(B60,8)
 *
  IODEVICE CUNUMBR=061,UNIT=2305,ADDRESS=(0F0,8)
  IODEVICE CUNUMBR=062,UNIT=2305,ADDRESS=(1F0,8)
  IODEVICE CUNUMBR=063,UNIT=2305,ADDRESS=(2F0,8)
  IODEVICE CUNUMBR=064,UNIT=2305,ADDRESS=(3F0,8)
  IODEVICE CUNUMBR=065,UNIT=2305,ADDRESS=(4F0,8)
  IODEVICE CUNUMBR=066,UNIT=2305,ADDRESS=(5F0,8)
```

```
  IODEVICE CUNUMBR=067,UNIT=2305,ADDRESS=(6F0,8)
  IODEVICE CUNUMBR=068,UNIT=2305,ADDRESS=(7F0,8)
  IODEVICE CUNUMBR=069,UNIT=2305,ADDRESS=(8F0,8)
  IODEVICE CUNUMBR=06A,UNIT=2305,ADDRESS=(9F0,8)
  IODEVICE CUNUMBR=06B,UNIT=2305,ADDRESS=(AF0,8)
  IODEVICE CUNUMBR=06C,UNIT=2305,ADDRESS=(BF0,8)
 *
  IODEVICE CUNUMBR=06D,UNIT=2305,ADDRESS=(0D0,8)
  IODEVICE CUNUMBR=06E,UNIT=2305,ADDRESS=(1D0,8)
  IODEVICE CUNUMBR=06F,UNIT=2305,ADDRESS=(2D0,8)
  IODEVICE CUNUMBR=070,UNIT=2305,ADDRESS=(3D0,8)
  IODEVICE CUNUMBR=071,UNIT=2305,ADDRESS=(4D0,8)
  IODEVICE CUNUMBR=072,UNIT=2305,ADDRESS=(5D0,8)
  IODEVICE CUNUMBR=073,UNIT=2305,ADDRESS=(6D0,8)
  IODEVICE CUNUMBR=074,UNIT=2305,ADDRESS=(7D0,8)
  IODEVICE CUNUMBR=075,UNIT=2305,ADDRESS=(8D0,8)
  IODEVICE CUNUMBR=076,UNIT=2305,ADDRESS=(9D0,8)
  IODEVICE CUNUMBR=077,UNIT=2305,ADDRESS=(AD0,8)
  IODEVICE CUNUMBR=078,UNIT=2305,ADDRESS=(BD0,8)
 *
  IODEVICE CUNUMBR=079,UNIT=3800,ADDRESS=(018,1)
  IODEVICE CUNUMBR=07A,UNIT=3800,ADDRESS=(118,1)
  IODEVICE CUNUMBR=07B,UNIT=3800,ADDRESS=(218,1)
  IODEVICE CUNUMBR=07C,UNIT=3800,ADDRESS=(318,1)
  IODEVICE CUNUMBR=07D,UNIT=3800,ADDRESS=(418,1)
  IODEVICE CUNUMBR=07E,UNIT=3800,ADDRESS=(518,1)
  IODEVICE CUNUMBR=07F,UNIT=3800,ADDRESS=(618,1)
 *
  IODEVICE CUNUMBR=080,UNIT=3278,ADDRESS=(020,8)
  IODEVICE CUNUMBR=081,UNIT=3278,ADDRESS=(120,8)
  IODEVICE CUNUMBR=082,UNIT=3278,ADDRESS=(220,8)
  IODEVICE CUNUMBR=083,UNIT=3278,ADDRESS=(320,8)
  IODEVICE CUNUMBR=084,UNIT=3278,ADDRESS=(420,8)
 *
  IODEVICE CUNUMBR=085,UNIT=1403,ADDRESS=(00E,1)
  IODEVICE CUNUMBR=086,UNIT=1403,ADDRESS=(10E,1)
  IODEVICE CUNUMBR=087,UNIT=1403,ADDRESS=(20E,1)
  IODEVICE CUNUMBR=088,UNIT=1403,ADDRESS=(30E,1)
  IODEVICE CUNUMBR=089,UNIT=1403,ADDRESS=(40E,1)
  IODEVICE CUNUMBR=08A,UNIT=1403,ADDRESS=(50E,1)
  IODEVICE CUNUMBR=08B,UNIT=1403,ADDRESS=(60E,1)
 *
  IODEVICE CUNUMBR=085,UNIT=1403,ADDRESS=(00F,1)
  IODEVICE CUNUMBR=086,UNIT=1403,ADDRESS=(10F,1)
  IODEVICE CUNUMBR=087,UNIT=1403,ADDRESS=(20F,1)
  IODEVICE CUNUMBR=088,UNIT=1403,ADDRESS=(30F,1)
  IODEVICE CUNUMBR=089,UNIT=1403,ADDRESS=(40F,1)
  IODEVICE CUNUMBR=08A,UNIT=1403,ADDRESS=(50F,1)
  IODEVICE CUNUMBR=08B,UNIT=1403,ADDRESS=(60F,1)
 *
  IODEVICE CUNUMBR=085,UNIT=2540R,ADDRESS=(00C,1)
  IODEVICE CUNUMBR=086,UNIT=2540R,ADDRESS=(10C,1)
  IODEVICE CUNUMBR=087,UNIT=2540R,ADDRESS=(20C,1)
  IODEVICE CUNUMBR=088,UNIT=2540R,ADDRESS=(30C,1)
  IODEVICE CUNUMBR=089,UNIT=2540R,ADDRESS=(40C,1)
  IODEVICE CUNUMBR=08A,UNIT=2540R,ADDRESS=(50C,1)
  IODEVICE CUNUMBR=08B,UNIT=2540R,ADDRESS=(60C,1)
 *
  IODEVICE CUNUMBR=085,UNIT=2540P,ADDRESS=(00D,1)
  IODEVICE CUNUMBR=086,UNIT=2540P,ADDRESS=(10D,1)
```

```
  IODEVICE CUNUMBR=087,UNIT=2540P,ADDRESS=(20D,1)
  IODEVICE CUNUMBR=088,UNIT=2540P,ADDRESS=(30D,1)
  IODEVICE CUNUMBR=089,UNIT=2540P,ADDRESS=(40D,1)
  IODEVICE CUNUMBR=08A,UNIT=2540P,ADDRESS=(50D,1)
  IODEVICE CUNUMBR=08B,UNIT=2540P,ADDRESS=(60D,1)
*
  IODEVICE CUNUMBR=08C,UNIT=3211,ADDRESS=(002,1)
  IODEVICE CUNUMBR=08D,UNIT=3211,ADDRESS=(102,1)
  IODEVICE CUNUMBR=08E,UNIT=3211,ADDRESS=(202,1)
  IODEVICE CUNUMBR=08F,UNIT=3211,ADDRESS=(302,1)
  IODEVICE CUNUMBR=090,UNIT=3211,ADDRESS=(402,1)
*
  IODEVICE CUNUMBR=091,UNIT=3278,ADDRESS=(040,8)
  IODEVICE CUNUMBR=092,UNIT=3278,ADDRESS=(140,8)
  IODEVICE CUNUMBR=093,UNIT=3278,ADDRESS=(240,8)
  IODEVICE CUNUMBR=094,UNIT=3278,ADDRESS=(340,8)
  IODEVICE CUNUMBR=095,UNIT=3278,ADDRESS=(440,8)
  IODEVICE CUNUMBR=096,UNIT=3278,ADDRESS=(540,8)
  IODEVICE CUNUMBR=097,UNIT=3278,ADDRESS=(640,8)
*
  IODEVICE CUNUMBR=098,UNIT=3505,ADDRESS=(012,1)
  IODEVICE CUNUMBR=099,UNIT=3505,ADDRESS=(112,1)
  IODEVICE CUNUMBR=09A,UNIT=3505,ADDRESS=(212,1)
  IODEVICE CUNUMBR=09B,UNIT=3505,ADDRESS=(312,1)
  IODEVICE CUNUMBR=09C,UNIT=3505,ADDRESS=(412,1)
*
  IODEVICE CUNUMBR=098,UNIT=3525,ADDRESS=(013,1)
  IODEVICE CUNUMBR=099,UNIT=3525,ADDRESS=(113,1)
  IODEVICE CUNUMBR=09A,UNIT=3525,ADDRESS=(213,1)
  IODEVICE CUNUMBR=09B,UNIT=3525,ADDRESS=(313,1)
  IODEVICE CUNUMBR=09C,UNIT=3525,ADDRESS=(413,1)
*
  IODEVICE CUNUMBR=09D,UNIT=3375,ADDRESS=(068,8)
  IODEVICE CUNUMBR=09E,UNIT=3375,ADDRESS=(168,8)
  IODEVICE CUNUMBR=09F,UNIT=3375,ADDRESS=(268,8)
  IODEVICE CUNUMBR=0A0,UNIT=3375,ADDRESS=(368,8)
  IODEVICE CUNUMBR=0A1,UNIT=3375,ADDRESS=(468,8)
  IODEVICE CUNUMBR=0A2,UNIT=3375,ADDRESS=(568,8)
  IODEVICE CUNUMBR=0A3,UNIT=3375,ADDRESS=(668,8)
*
  IODEVICE CUNUMBR=0A4,UNIT=3375,ADDRESS=(090,8)
  IODEVICE CUNUMBR=0A5,UNIT=3375,ADDRESS=(190,8)
  IODEVICE CUNUMBR=0A6,UNIT=3375,ADDRESS=(290,8)
  IODEVICE CUNUMBR=0A7,UNIT=3375,ADDRESS=(390,8)
  IODEVICE CUNUMBR=0A8,UNIT=3375,ADDRESS=(490,8)
*
  IODEVICE CUNUMBR=0A9,UNIT=3380,ADDRESS=(038,8)
  IODEVICE CUNUMBR=0AA,UNIT=3380,ADDRESS=(138,8)
  IODEVICE CUNUMBR=0AB,UNIT=3380,ADDRESS=(238,8)
  IODEVICE CUNUMBR=0AC,UNIT=3380,ADDRESS=(338,8)
  IODEVICE CUNUMBR=0AD,UNIT=3380,ADDRESS=(438,8)
  IODEVICE CUNUMBR=0AE,UNIT=3380,ADDRESS=(538,8)
  IODEVICE CUNUMBR=0AF,UNIT=3380,ADDRESS=(638,8)
*
  IODEVICE CUNUMBR=0B0,UNIT=3380,ADDRESS=(0A0,8)
  IODEVICE CUNUMBR=0B1,UNIT=3380,ADDRESS=(1A0,8)
  IODEVICE CUNUMBR=0B2,UNIT=3380,ADDRESS=(2A0,8)
  IODEVICE CUNUMBR=0B3,UNIT=3380,ADDRESS=(3A0,8)
  IODEVICE CUNUMBR=0B4,UNIT=3380,ADDRESS=(4A0,8)
*
```

```
  IODEVICE CUNUMBR=0B5,UNIT=3203,ADDRESS=(003,1)
  IODEVICE CUNUMBR=0B6,UNIT=3203,ADDRESS=(103,1)
  IODEVICE CUNUMBR=0B7,UNIT=3203,ADDRESS=(203,1)
  IODEVICE CUNUMBR=0B8,UNIT=3203,ADDRESS=(303,1)
*
  IODEVICE CUNUMBR=0B9,UNIT=3480,ADDRESS=(004,4)
  IODEVICE CUNUMBR=0BA,UNIT=3480,ADDRESS=(104,4)
  IODEVICE CUNUMBR=0BB,UNIT=3480,ADDRESS=(204,4)
  IODEVICE CUNUMBR=0BC,UNIT=3480,ADDRESS=(304,4)
*
  IODEVICE CUNUMBR=0BD,UNIT=3890,ADDRESS=(008,4)
  IODEVICE CUNUMBR=0BE,UNIT=3890,ADDRESS=(108,4)
*
*  5080 UNIT
*
  IODEVICE CUNUMBR=0BF,UNIT=HFGD,ADDRESS=(014,4)
  IODEVICE CUNUMBR=0C0,UNIT=HFGD,ADDRESS=(114,4)
*
```

Sample Files for a 3350 Starter System

IBM provides the following sample files with the VM/XA SP 3350 starter system:

- HCPSYS ASSEMBLE for starter system
- System directory
- System residence DASD allocation
- Minidisk maps of the system residence device.

Sample HCPSYS ASSEMBLE for 3350

This is only a sample file. You need to tailor it to your installation before it can be used.

```
SYS TITLE 'HCPSYS50 - HCPSYS FOR VM/XA INSTALL BUILD'
*****************************************************************
***     Virtual Machine / System Product       5664-308    ***
***     Copyright (c) I B M   Corporation      1988, 1989  ***
***     Licensed Materials -  Property of I B M            ***
***     Refer to Copyright Instructions: Form G120-2083    ***
*****************************************************************
*****************************************************************
*
* MODULE NAME - HCPSYS
*
* DESCRIPTIVE NAME - DEFINITION OF SYSTEM
*
*
* FUNCTION - TO DEFINE THE SYSTEM
*
*
*   REGISTER CONVENTIONS - SYMBOLIC REFERENCES TO REGISTERS ARE
*                          OF THE FORM "RX" WHERE X IS A NUMBER
*                          RANGING FROM 0 TO 15 (SEE HCPEQUAT COPY)
*
*   PATCH LABEL - NONE
*
* MODULE TYPE - DATA
*
*   PROCESSOR - ASSEMBLER H (VERSION 2)
*
*   ATTRIBUTES - RESIDENT, DATA-ONLY
*
* MACROS -
*
*          SYSSTORE - DEFINE SYSTEM REAL STORAGE
*          SYSRES   - DEFINE SYSTEM RESIDENCE VOLUME
*          SYSCPVOL - DEFINE SYSTEM VOLUMES
*          SYSOPR   - DEFINE SYSTEM OPERATOR
*          SYSTIME  - DEFINE THE TIME ZONE
*          SYSACNT  - DEFINE ACCOUNTING USERID
*          SYSDUMP  - DEFINE DUMP RECEIVER USERID
*          SYSEREP  - DEFINE EREP RECEIVER USERID
*          SYSID    - DEFINE SYSTEM ID'S BY PROCESSOR
*          SYSUVOL  - DEFINE USER VOLUMES TO BE MOUNTED
*          SYSFCN   - DEFINE PRIV CLASSES ASSIGNED TO CP FUNCTIONS
*          SYSEND   - END SYSTEM GENERATION
*
*
```

```
* GENERAL COMMENTS - THIS MODULE IS THE USER'S RESPONSIBILITY.
*
*********************************************************************
         SPACE 1
*-------------------------------------------------------------
* THE SYSTEM DEFINITION MACROS SYSSTORE, SYSRES, SYSCPVOL,
* SYSOPR, AND SYSTIME, ARE CODED HERE, IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSSTORE RMSIZE=32M,VRSIZE=8M,VRFREE=256
         SYSRES SYSVOL=XASRES,                                  *
               SYSRES=123,SYSTYPE=3350,                         *
               SYSNUC=003,                                      *
               SYSCKP=(013,2),                                  *
               SYSWRM=(015,2)
         SYSTIME ZONE=5,LOC=WEST,ID=EST
         SYSOPR SYSOPER=OPERATOR
         SYSCPVOL     XASRES,XASERV,XAP001,XAP002,              *
               DRUM01,DRUM02,DRUM03,DRUM04,DRUM05,DRUM06,DRUM07,*
               PAG001,PAG002,PAG003,PAG004,PAG005,PAG006,PAG007,*
               SPL001,SPL002,SPL003,SPL004,SPL005,SPL006,SPL007
         SPACE 1
*-------------------------------------------------------------
* ANY OF THE OPTIONAL MACROS THAT ARE DESIRED ARE CODED HERE,
* IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSID      DEFAULT=INSTTST,(3081A,3081,10000)
         SYSACNT USERID=DISKACNT
         SYSDUMP USERID=OPERATNS
         SYSEREP USERID=EREP
         SYSUVOL      USRP01,USRP02,USRP03,USRP04,USRP05,USRP06,USRP07,*
               USRP08,USRP09
         SYSFCN  OPER=A,CPRD=CE,CPWT=C,SERV=F,DFLT=G
         SPACE 1
*-------------------------------------------------------------
* THE SYSEND MACRO IS CODED HERE
*-------------------------------------------------------------
         SPACE 1
         SYSEND
```

Sample Directory for a 3350

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1988, 1989 *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM    *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083         *
*************************************************************
*************************************************************
*  3350 SYSTEM   VM/XA SYSTEM  PRODUCT  DIRECTORY           *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.   *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST  *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,   *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.   *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL  *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH   *
*  YOUR HARDWARE ADDRESSES.                                 *
*                                                           *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO  *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE  *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY  *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                 *
*                                                           *
*      USER ANYONE SOMEPASS 3M 8M FG                        *
*       AUTOLOG AUTOLOG1 OP1 MAINT                          *
*        ACCOUNT ??????                                     *
*        IPL 190                                            *
*        CONSOLE 01F 3215                                   *
*        SPOOL 00C 2540 READER A                            *
*        SPOOL 00D 2540 PUNCH A                             *
*        SPOOL 00E 1403 A                                   *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS  *
*                                                           *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                READ ONLY MODE.                            *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                WRITE MODE.                                *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                MULTI-WRITE MODE.                          *
*                                                           *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.   *
*                                                           *
*************************************************************
*                                                           *
*  Following is a sample user entry and a brief description  *
*  of each entry.  A more detailed description may be found  *
*  by referencing the VM/XA System Product Planning and     *
*  Administration manual.                                   *
*                                                           *
*    USER CMS1  abc123  3M  32M G                           *
*       - Userid is CMS1                                    *
*       - Logon password is abc123                          *
*       - At logon time, virtual storage will equal 3M      *
*       - Maximum storage that may be defined equals 32M    *
*       - Privilege class is G                              *
*                                                           *
*     AUTOLOG AUTOLOG1 OP1 MAINT                            *
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,     *
```

```
*         OP1, and MAINT                                        *
*                                                               *
*     ACCOUNT ACT4 CMSTST                                       *
*       - Time used by CMS1 will be charged to account ACT4     *
*       - Printed output will be sent to distribution CMSTST    *
*                                                               *
*     IPL CMS                                                   *
*       - Will cause automatic IPL of named save system of      *
*         CMS                                                   *
*                                                               *
*     CONSOLE 009 3215                                          *
*       - Defines virtual machines console as a 3215 at a       *
*         virtual address of 009                                *
*                                                               *
*     SPOOL 00C 2540 READER A                                   *
*     SPOOL 00D 2540 PUNCH A                                    *
*     SPOOL 00E 1403 A                                          *
*       - Defines virtual unit record devices of reader,        *
*         punch, and printer with a spooling class of 'A'       *
*                                                               *
*     LINK MAINT 190 190 RR                                     *
*       - Allows read access to minidisk 190 of user 'MAINT'    *
*                                                               *
*     LINK MAINT 19E 19E RR                                     *
*       - Allows read access to minidisk 19E of user 'MAINT'    *
*                                                               *
*     LINK MAINT 19D 19D RR                                     *
*       - Allows read access to minidisk 19d of user 'MAINT'    *
*                                                               *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS              *
*       - Defines minidisk with a virtual address of 999        *
*       - Specifies that the minidisk resides on a real 3350    *
*       - Minidisk starts at cylinder 236                       *
*       - Mdisk is 22 cylinders in size (starting at cyl 236    *
*       - The real volume serial number is 'CPPACK'             *
*       - Mdisk is to be accessed in write mode if no other     *
*         user has write access. Alternate access read-only     *
*       - RPASS is the required password for another user to    *
*         link to this minidisk in read mode                    *
*       - WPASS is the required password for another user to    *
*         link to this minidisk in write mode                   *
*                                                               *
*****************************************************************
*
*
*
DIRECTORY 123 3350 XASRES
*
*****************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS               *
*****************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3350 000 001 XASRES  R
 MDISK B01 3350 000 001 XASERV  R
 MDISK C01 3350 000 001 XAP001  R
 MDISK D01 3350 000 001 XAP002  R
*
USER $DIRECT$ NOLOG
```

```
 MDISK A04 3350 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3350 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3350 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3350 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3350 017 025 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3350 042 060 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3350 306 020 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                  PROP LOGICAL OPERATOR                            *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3350 253 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
```

```
NAMESAVE GCS
NAMESAVE VTAM
NAMESAVE HELP
NAMESAVE INSTHELP
CONSOLE 009 3215 T
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
MDISK 123 3350 000 555 XASRES  MW
MDISK 124 3350 000 555 XAP001  RR
MDISK 125 3350 000 555 XASERV  RR
MDISK 126 3350 000 555 XAP002  RR

MDISK 191 3350 326 039 XASRES  MW
MDISK 193 3350 414 060 XASRES  MW
MDISK 194 3350 001 065 XASERV  MW
MDISK 201 3350 066 028 XASERV  MW RMAINT   WMAINT   MMAINT

MDISK 292 3350 094 025 XASERV  MW READ WRITE MULTIPLE
MDISK 392 3350 119 025 XASERV  MW READ WRITE MULTIPLE
MDISK 492 3350 157 013 XASERV  MW READ WRITE MULTIPLE
MDISK 692 3350 144 013 XASERV  MW READ WRITE MULTIPLE
MDISK 593 3350 170 113 XASERV  MW READ WRITE MULTIPLE
MDISK 594 3350 406 075 XAP002  MW READ WRITE MULTIPLE

MDISK 295 3350 474 018 XASRES  MW READ WRITE MULTIPLE
MDISK 395 3350 512 013 XASRES  MW READ WRITE MULTIPLE
MDISK 495 3350 357 007 XASERV  MW READ WRITE MULTIPLE
MDISK 592 3350 344 013 XASERV  MW READ WRITE MULTIPLE
MDISK 892 3350 364 007 XASERV  MW READ WRITE MULTIPLE
MDISK 491 3350 334 010 XASERV  MW READ WRITE MULTIPLE
MDISK 293 3350 371 175 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
MDISK 294 3350 164 150 XAP001  MW RCPAUX   WCPAUX   MCPAUX
MDISK 393 3350 326 080 XAP002  WR
MDISK 394 3350 001 163 XAP001  WR
MDISK 19C 3350 256 050 XASRES  WR
MDISK 49C 3350 481 050 XAP002  WR

MDISK 190 3350 123 074 XASRES  MW ALL
MDISK 791 3350 197 015 XASRES  MW READ WRITE MULTIPLE
MDISK 89E 3350 102 013 XASRES  MW READ WRITE MULTIPLE
MDISK 5E5 3350 115 007 XASRES  MW READ WRITE MULTIPLE
MDISK 896 3350 227 013 XASRES  MW READ WRITE MULTIPLE
MDISK 895 3350 240 013 XASRES  MW READ WRITE MULTIPLE
MDISK 19E 3350 314 055 XAP001  MW ALL
MDISK 19D 3350 284 050 XASERV  MW ALL

MDISK 291 3350 369 019 XAP001  MW
MDISK 391 3350 388 019 XAP001  MW
MDISK 591 3350 001 119 XAP002  MW READ WRITE MULTIPLE
MDISK 691 3350 120 138 XAP002  MW READ WRITE MULTIPLE
MDISK 49D 3350 500 050 XAP001  MW READ WRITE MULTIPLE
MDISK 501 3350 481 003 XAP001  MW READ WRITE MULTIPLE
MDISK 595 3350 484 016 XAP001  MW READ WRITE MULTIPLE
*
MDISK 490 3350 407 074 XAP001  MW
MDISK 423 3350 272 013 XAP002  MW
MDISK 596 3350 285 025 XAP002  MW
```

```
 MDISK 59E 3350 310 013 XAP002  MW
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3350 323 003 XAP002 RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3350 254 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3350 223 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT   NOLOG  3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR   NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3350 386 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
```

```
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3350 391 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3350 000 555 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
```

```
  MDISK 191 3350 412 002 XASRES WR READ     WRITE     MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3350 392 017 XASRES MR RDVF     WDVF     MDVF
*
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1   NOLOG  3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3350 409 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3350 410 001 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
```

```
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3350 411 001 XASRES MR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3350 000 555 XASRES RR
 MDISK 124 3350 000 555 XAP001 RR
 MDISK 125 3350 000 555 XASERV RR
 MDISK 126 3350 000 555 XAP002 RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
```

```
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM     NOLOG    3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3350

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3350 000 555 XASRES  MW
 MDISK 124 3350 000 555 XAP001  RR
 MDISK 125 3350 000 555 XASERV  RR
 MDISK 126 3350 000 555 XAP002  RR

 MDISK 191 3350 326 039 XASRES  MW
 MDISK 193 3350 414 060 XASRES  MW
 MDISK 194 3350 001 065 XASERV  MW
 MDISK 201 3350 066 028 XASERV  MW RMAINT   WMAINT   MMAINT

 MDISK 292 3350 094 025 XASERV  MW READ WRITE MULTIPLE
 MDISK 392 3350 119 025 XASERV  MW READ WRITE MULTIPLE
 MDISK 492 3350 157 013 XASERV  MW READ WRITE MULTIPLE
 MDISK 692 3350 144 013 XASERV  MW READ WRITE MULTIPLE
 MDISK 593 3350 170 113 XASERV  MW READ WRITE MULTIPLE
 MDISK 594 3350 406 075 XAP002  MW READ WRITE MULTIPLE

 MDISK 295 3350 474 018 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3350 512 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3350 357 007 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3350 344 013 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3350 364 007 XASERV  MW READ WRITE MULTIPLE
 MDISK 491 3350 334 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 293 3350 371 175 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3350 164 150 XAP001  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 393 3350 326 080 XAP002  WR
 MDISK 394 3350 001 163 XAP001  WR
 MDISK 19C 3350 256 050 XASRES  WR
 MDISK 49C 3350 481 050 XAP002  WR

 MDISK 190 3350 123 074 XASRES  MW ALL
 MDISK 791 3350 197 015 XASRES  MW READ WRITE MULTIPLE
 MDISK 89E 3350 102 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 5E5 3350 115 007 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3350 227 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 895 3350 240 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 19E 3350 314 055 XAP001  MW ALL
 MDISK 19D 3350 284 050 XASERV  MW ALL

 MDISK 291 3350 369 019 XAP001  MW
 MDISK 391 3350 388 019 XAP001  MW
```

```
 MDISK 591 3350 001 119 XAP002  MW READ WRITE MULTIPLE
 MDISK 691 3350 120 138 XAP002  MW READ WRITE MULTIPLE
 MDISK 49D 3350 500 050 XAP001  MW READ WRITE MULTIPLE
 MDISK 501 3350 481 003 XAP001  MW READ WRITE MULTIPLE
 MDISK 595 3350 484 016 XAP001  MW READ WRITE MULTIPLE
*
 MDISK 490 3350 407 074 XAP001  MW
 MDISK 423 3350 272 013 XAP002  MW
 MDISK 596 3350 285 025 XAP002  MW
 MDISK 59E 3350 310 013 XAP002  MW
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3350 323 003 XAP002 RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
 LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3350

Figure 54 shows the allocation for the system residence device for the 3350 VM/XA SP Starter System.

```
TYPE      CYL  CYL
PERM      0000-0000
DRCT      0001-0002
PERM      0003-0016
PAGE      0017-0041
SPOL      0042-0101
PERM      0102-0305
TDSK      0306-0325
PERM      0326-0554
```

Figure 54. System Residence DASD Allocation for a 3350

Minidisk Maps for a 3350 System Residence Device

Figure 55 shows the minidisk maps for the system residence device for the 3350. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3350     0000     0554     0555

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3350     0000     0000     0001
         MAINT       394     3350     0001     0163     0163
         MAINT       294     3350     0164     0313     0150
         MAINT       19E     3350     0314     0368     0055
         MAINT       291     3350     0369     0387     0019
         MAINT       391     3350     0388     0406     0019
         MAINT       490     3350     0407     0480     0074
         MAINT       501     3350     0481     0483     0003
         MAINT       595     3350     0484     0499     0016
         MAINT       49D     3350     0500     0549     0050
         MAINT       124     3350     0000     0554     0555 **OVERLAP**
         SYSDUMP1    124     3350     0000     0554     0555 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP002   $ALLOC$     D01     3350     0000     0000     0001
         MAINT       591     3350     0001     0119     0119
         MAINT       691     3350     0120     0257     0138
                                       258      271       14    GAP
         MAINT       423     3350     0272     0284     0013
         MAINT       596     3350     0285     0309     0025
         MAINT       59E     3350     0310     0322     0013
         MAINT       36E     3350     0323     0325     0003
         MAINT       393     3350     0326     0405     0080
         MAINT       594     3350     0406     0480     0075
         MAINT       49C     3350     0481     0530     0050
         MAINT       126     3350     0000     0554     0555 **OVERLAP**
         SYSDUMP1    126     3350     0000     0554     0555 **OVERLAP**
         GCS         191     3350     0223     0226     0004
         MAINT       896     3350     0227     0239     0013
         MAINT       895     3350     0240     0252     0013
         LGLOPR      191     3350     0253     0253     0001
         CMSBATCH    195     3350     0254     0255     0002
         MAINT       19C     3350     0256     0305     0050
         $T-DISK$    B01     3350     0306     0325     0020
         MAINT       191     3350     0326     0364     0039
                                       365      385       21    GAP
```

Figure 55 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3350

```
         OPERATOR   191    3350    0386    0390    0005
         OP1        191    3350    0391    0391    0001
         OPERATNS   191    3350    0392    0408    0017
         AUTOLOG1   191    3350    0409    0409    0001
         DISKACNT   191    3350    0410    0410    0001
         IBMPSR     191    3350    0411    0411    0001
         EREP       191    3350    0412    0413    0002
         MAINT      193    3350    0414    0473    0060
         MAINT      295    3350    0474    0491    0018
                                     492     511      20    GAP
         MAINT      395    3350    0512    0524    0013
         MAINT      123    3350    0000    0554    0555 **OVERLAP**
         SYSDUMP1   123    3350    0000    0554    0555 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID     CUU   DEVTYPE  START     END     SIZE
XASERV   $ALLOC$    B01     3350    0000    0000    0001
         MAINT      194     3350    0001    0065    0065
         MAINT      201     3350    0066    0093    0028
         MAINT      292     3350    0094    0118    0025
         MAINT      392     3350    0119    0143    0025
         MAINT      692     3350    0144    0156    0013
         MAINT      492     3350    0157    0169    0013
         MAINT      593     3350    0170    0282    0113
                                     283     283       1    GAP
         MAINT      19D     3350    0284    0333    0050
         MAINT      491     3350    0334    0343    0010
         MAINT      592     3350    0344    0356    0013
         MAINT      495     3350    0357    0363    0007
         MAINT      892     3350    0364    0370    0007
         MAINT      293     3350    0371    0545    0175
         MAINT      125     3350    0000    0554    0555 **OVERLAP**
         SYSDUMP1   125     3350    0000    0554    0555 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID     CUU   DEVTYPE  START     END     SIZE
XASRES   $ALLOC$    A01     3350    0000    0000    0001
         $DIRECT$   A04     3350    0001    0002    0002
         $CP-NUC$   A09     3350    0003    0012    0010
         $SYSCKP$   A06     3350    0013    0014    0002
         $SYSWRM$   A07     3350    0015    0016    0002
         $PAGE$     A03     3350    0017    0041    0025
         $SPOOL$    B01     3350    0042    0101    0060
         MAINT      89E     3350    0102    0114    0013
         MAINT      5E5     3350    0115    0121    0007
                                     122     122       1    GAP
         MAINT      190     3350    0123    0196    0074
         MAINT      791     3350    0197    0211    0015
                                     212     222      11    GAP
```

Figure 55 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3350

Sample Files for a 3375 Starter System

IBM provides the following sample files with the VM/XA System Product 3375 starter system:

- HCPSYS ASSEMBLE for starter system
- System directory
- System residence DASD allocation
- Minidisk maps of the system residence device.

Sample HCPSYS ASSEMBLE for a 3375

This is only a sample file. You need to tailor it to your installation before it can be used.

```
SYS TITLE 'HCPSYS75 - HCPSYS FOR VM/XA INSTALL BUILD'
*****************************************************************
***     Virtual Machine / System Product       5664-308    ***
***     Copyright (c) I B M   Corporation      1988, 1989  ***
***     Licensed Materials -  Property of I B M            ***
***     Refer to Copyright Instructions: Form G120-2083    ***
*****************************************************************
*****************************************************************
*
* MODULE NAME - HCPSYS
*
* DESCRIPTIVE NAME - DEFINITION OF SYSTEM
*
*
* FUNCTION - TO DEFINE THE SYSTEM
*
*
*   REGISTER CONVENTIONS - SYMBOLIC REFERENCES TO REGISTERS ARE
*                          OF THE FORM "RX" WHERE X IS A NUMBER
*                          RANGING FROM 0 TO 15 (SEE HCPEQUAT COPY)
*
*   PATCH LABEL - NONE
*
* MODULE TYPE - DATA
*
*   PROCESSOR - ASSEMBLER H (VERSION 2)
*
*   ATTRIBUTES - RESIDENT, DATA-ONLY
*
* MACROS -
*
*          SYSSTORE - DEFINE SYSTEM REAL STORAGE
*          SYSRES   - DEFINE SYSTEM RESIDENCE VOLUME
*          SYSCPVOL - DEFINE SYSTEM VOLUMES
*          SYSOPR   - DEFINE SYSTEM OPERATOR
*          SYSTIME  - DEFINE THE TIME ZONE
*          SYSACNT  - DEFINE ACCOUNTING USERID
*          SYSDUMP  - DEFINE DUMP RECEIVER USERID
*          SYSEREP  - DEFINE EREP RECEIVER USERID
*          SYSID    - DEFINE SYSTEM ID'S BY PROCESSOR
*          SYSUVOL  - DEFINE USER VOLUMES TO BE MOUNTED
*          SYSFCN   - DEFINE PRIV CLASSES ASSIGNED TO CP FUNCTIONS
*          SYSEND   - END SYSTEM GENERATION
*
*
```

```
* GENERAL COMMENTS - THIS MODULE IS THE USER'S RESPONSIBILITY.
*
*********************************************************************
         SPACE 1
*-------------------------------------------------------------
* THE SYSTEM DEFINITION MACROS SYSSTORE, SYSRES, SYSCPVOL,
* SYSOPR, AND SYSTIME, ARE CODED HERE, IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSSTORE RMSIZE=32M,VRSIZE=8M,VRFREE=256
         SYSRES SYSVOL=XASRES,                                  *
               SYSRES=123,SYSTYPE=3375,                         *
               SYSNUC=005,                                      *
               SYSCKP=(021,4),                                  *
               SYSWRM=(025,4)
         SYSTIME ZONE=5,LOC=WEST,ID=EST
         SYSOPR SYSOPER=OPERATOR
         SYSCPVOL     XASRES,XASERV,XAP001,                     *
               DRUM01,DRUM02,DRUM03,DRUM04,DRUM05,DRUM06,DRUM07,*
               PAG001,PAG002,PAG003,PAG004,PAG005,PAG006,PAG007,*
               SPL001,SPL002,SPL003,SPL004,SPL005,SPL006,SPL007
         SPACE 1
*-------------------------------------------------------------
* ANY OF THE OPTIONAL MACROS THAT ARE DESIRED ARE CODED HERE,
* IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSID      DEFAULT=INSTTST,(3081A,3081,10000)
         SYSACNT USERID=DISKACNT
         SYSDUMP USERID=OPERATNS
         SYSEREP USERID=EREP
         SYSUVOL      USRP01,USRP02,USRP03,USRP04,USRP05,USRP06,USRP07,*
               USRP08,USRP09
         SYSFCN  OPER=A,CPRD=CE,CPWT=C,SERV=F,DFLT=G
         SPACE 1
*-------------------------------------------------------------
* THE SYSEND MACRO IS CODED HERE
*-------------------------------------------------------------
         SPACE 1
         SYSEND
```

Sample Directory for a 3375

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*****************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1988, 1989 *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM    *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083         *
*****************************************************************
*****************************************************************
*  3375 SYSTEM   VM/XA SYSTEM  PRODUCT  DIRECTORY               *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.       *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND   *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST    *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,       *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.       *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL     *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH       *
*  YOUR HARDWARE ADDRESSES.                                     *
*                                                               *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO    *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE     *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY     *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                     *
*                                                               *
*      USER ANYONE SOMEPASS 3M 8M FG                            *
*       AUTOLOG AUTOLOG1 OP1 MAINT                              *
*        ACCOUNT ??????                                         *
*        IPL 190                                                *
*        CONSOLE 01F 3215                                       *
*        SPOOL 00C 2540 READER A                                *
*        SPOOL 00D 2540 PUNCH A                                 *
*        SPOOL 00E 1403 A                                       *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS     *
*                                                               *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                READ ONLY MODE.                                *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                WRITE MODE.                                    *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                MULTI-WRITE MODE.                              *
*                                                               *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.       *
*                                                               *
*                                                               *
*****************************************************************
*                                                               *
*  Following is a sample user entry and a brief description     *
*  of each entry.  A more detailed description may be found     *
*  by referencing the VM/XA System Product Planning and         *
*  Administration manual.                                       *
*                                                               *
*    USER CMS1  abc123  3M  32M G                               *
*       - Userid is CMS1                                        *
*       - Logon password is abc123                              *
*       - At logon time, virtual storage will equal 3M          *
*       - Maximum storage that may be defined equals 32M        *
*       - Privilege class is G                                  *
*                                                               *
*     AUTOLOG AUTOLOG1 OP1 MAINT                                *
```

```
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,      *
*         OP1, and MAINT                                     *
*                                                            *
*     ACCOUNT ACT4 CMSTST                                    *
*       - Time used by CMS1 will be charged to account ACT4  *
*       - Printed output will be sent to distribution CMSTST *
*                                                            *
*     IPL CMS                                                *
*       - Will cause automatic IPL of named save system of   *
*         CMS                                                *
*                                                            *
*     CONSOLE 009 3215                                       *
*       - Defines virtual machines console as a 3215 at a    *
*         virtual address of 009                             *
*                                                            *
*     SPOOL 00C 2540 READER A                                *
*     SPOOL 00D 2540 PUNCH A                                 *
*     SPOOL 00E 1403 A                                       *
*       - Defines virtual unit record devices of reader,     *
*         punch, and printer with a spooling class of 'A'    *
*                                                            *
*     LINK MAINT 190 190 RR                                  *
*       - Allows read access to minidisk 190 of user 'MAINT' *
*                                                            *
*     LINK MAINT 19E 19E RR                                  *
*       - Allows read access to minidisk 19E of user 'MAINT' *
*                                                            *
*     LINK MAINT 19D 19D RR                                  *
*       - Allows read access to minidisk 19d of user 'MAINT' *
*                                                            *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS           *
*       - Defines minidisk with a virtual address of 999     *
*       - Specifies that the minidisk resides on a real 3350 *
*       - Minidisk starts at cylinder 236                    *
*       - Mdisk is 22 cylinders in size (starting at cyl 236 *
*       - The real volume serial number is 'CPPACK'          *
*       - Mdisk is to be accessed in write mode if no other  *
*         user has write access. Alternate access read-only  *
*       - RPASS is the required password for another user to *
*         link to this minidisk in read mode                 *
*       - WPASS is the required password for another user to *
*         link to this minidisk in write mode                *
**************************************************************
*
*
*
DIRECTORY 123 3375 XASRES
*
**************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS            *
**************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3375 000 001 XASRES R
 MDISK B01 3375 000 001 XASERV R
 MDISK C01 3375 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
```

```
 MDISK A04 3375 001 004 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3375 005 016 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3375 021 004 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3375 025 004 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3375 029 038 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3375 067 092 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3375 561 031 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                  PROP LOGICAL OPERATOR                            *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3375 487 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
```

```
 NAMESAVE GCS
 NAMESAVE VTAM
 NAMESAVE HELP
 NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3375 000 959 XASRES  MW
 MDISK 124 3375 000 959 XAP001  RR
 MDISK 125 3375 000 959 XASERV  RR
*
 MDISK 191 3375 492 047 XASRES  MW
 MDISK 193 3375 742 076 XASRES  MW
 MDISK 194 3375 818 082 XASRES  MW
 MDISK 201 3375 064 035 XASERV  MW RMAINT   WMAINT   MMAINT
*
 MDISK 292 3375 099 031 XASERV  MW READ WRITE MULTIPLE
 MDISK 392 3375 130 031 XASERV  MW READ WRITE MULTIPLE
 MDISK 492 3375 161 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 692 3375 177 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 594 3375 193 118 XASERV  MW READ WRITE MULTIPLE
 MDISK 593 3375 327 141 XASERV  MW READ WRITE MULTIPLE
 MDISK 423 3375 311 016 XASERV  MW
*
 MDISK 295 3375 676 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3375 447 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3375 468 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3375 476 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3375 492 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 491 3375 500 013 XASERV  MW READ WRITE MULTIPLE
 MDISK 791 3375 513 019 XASERV  MW
 MDISK 89E 3375 592 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 5E5 3375 423 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3375 608 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 895 3375 624 016 XASRES  MW READ WRITE MULTIPLE

 MDISK 293 3375 532 220 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3375 752 188 XASERV  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 393 3375 001 098 XAP001  WR
 MDISK 394 3375 743 204 XAP001  WR
 MDISK 19C 3375 103 063 XAP001  WR
 MDISK 49C 3375 166 063 XAP001  WR
 MDISK 596 3375 183 032 XASRES  WR
 MDISK 59E 3375 463 016 XASRES  WR
*
 MDISK 190 3375 223 113 XASRES  MW ALL
 MDISK 19E 3375 235 069 XAP001  MW ALL
 MDISK 19D 3375 001 063 XASERV  MW ALL
*
 MDISK 291 3375 336 024 XASRES  MW
 MDISK 391 3375 159 024 XASRES  MW
 MDISK 591 3375 304 149 XAP001  MW READ WRITE MULTIPLE
 MDISK 691 3375 453 173 XAP001  MW READ WRITE MULTIPLE
 MDISK 49D 3375 360 063 XASRES  MW READ WRITE MULTIPLE
 MDISK 490 3375 626 113 XAP001  MW
 MDISK 501 3375 739 004 XAP001  MW READ WRITE MULTIPLE
 MDISK 595 3375 640 020 XASRES  MW READ WRITE MULTIPLE
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
```

```
 MDISK 36E 3375 099 004 XAP001 RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3375 488 004 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3375 441 006 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT  NOLOG   3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3375 696 008 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
```

```
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3375 704 002 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3375 000 959 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
  MDISK 191 3375 738 004 XASRES WR READ     WRITE   MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
```

```
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3375 706 026 XASRES MR RDVF     WDVF     MDVF
*
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3375 732 002 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3375 734 002 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
```

```
 MDISK 191 3375 736 002 XASRES WR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3375 000 959 XASRES RR
 MDISK 124 3375 000 959 XAP001 RR
 MDISK 125 3375 000 959 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
```

```
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3375

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3375 000 959 XASRES  MW
 MDISK 124 3375 000 959 XAP001  RR
 MDISK 125 3375 000 959 XASERV  RR
*
 MDISK 191 3375 492 047 XASRES  MW
 MDISK 193 3375 742 076 XASRES  MW
 MDISK 194 3375 818 082 XASRES  MW
 MDISK 201 3375 064 035 XASERV  MW RMAINT   WMAINT   MMAINT
*
 MDISK 292 3375 099 031 XASERV  MW READ WRITE MULTIPLE
 MDISK 392 3375 130 031 XASERV  MW READ WRITE MULTIPLE
 MDISK 492 3375 161 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 692 3375 177 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 594 3375 193 118 XASERV  MW READ WRITE MULTIPLE
 MDISK 593 3375 327 141 XASERV  MW READ WRITE MULTIPLE
 MDISK 423 3375 311 016 XASERV  MW
*
 MDISK 295 3375 676 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3375 447 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3375 468 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3375 476 016 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3375 492 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 491 3375 500 013 XASERV  MW READ WRITE MULTIPLE
 MDISK 791 3375 513 019 XASERV  MW
 MDISK 89E 3375 592 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 5E5 3375 423 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3375 608 016 XASRES  MW READ WRITE MULTIPLE
 MDISK 895 3375 624 016 XASRES  MW READ WRITE MULTIPLE

 MDISK 293 3375 532 220 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3375 752 188 XASERV  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 393 3375 001 098 XAP001  WR
 MDISK 394 3375 743 204 XAP001  WR
 MDISK 19C 3375 103 063 XAP001  WR
 MDISK 49C 3375 166 063 XAP001  WR
 MDISK 596 3375 183 032 XASRES  WR
 MDISK 59E 3375 463 016 XAP001  WR
*
 MDISK 190 3375 223 113 XASRES  MW ALL
 MDISK 19E 3375 235 069 XAP001  MW ALL
```

```
 MDISK 19D 3375 001 063 XASERV  MW ALL
*
 MDISK 291 3375 336 024 XASRES  MW
 MDISK 391 3375 159 024 XASRES  MW
 MDISK 591 3375 304 149 XAP001  MW READ WRITE MULTIPLE
 MDISK 691 3375 453 173 XAP001  MW READ WRITE MULTIPLE
 MDISK 49D 3375 360 063 XASRES  MW READ WRITE MULTIPLE
 MDISK 490 3375 626 113 XAP001  MW
 MDISK 501 3375 739 004 XAP001  MW READ WRITE MULTIPLE
 MDISK 595 3375 640 020 XASRES  MW READ WRITE MULTIPLE
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3375 099 004 XAP001 RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
 LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3375

Figure 56 shows the allocation for the system residence device for the 3375 VM/XA SP starter system.

```
TYPE      CYL  CYL
PERM      0000-0000
DRCT      0001-0004
PERM      0005-0028
PAGE      0029-0066
SPOL      0067-0158
PERM      0159-0560
TDSK      0561-0591
PERM      0592-0958
```

Figure 56. System Residence DASD Allocation for a 3375

Minidisk Maps for a 3375 System Residence Device

Figure 57 shows the minidisk maps of the system residence device for the 3375. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3375     0000     0958     0959

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3375     0000     0000     0001
         MAINT       393     3375     0001     0098     0098
         MAINT       36E     3375     0099     0102     0004
         MAINT       19C     3375     0103     0165     0063
         MAINT       49C     3375     0166     0228     0063
                                       229      234        6    GAP
         MAINT       19E     3375     0235     0303     0069
         MAINT       591     3375     0304     0452     0149
         MAINT       691     3375     0453     0625     0173
         MAINT       490     3375     0626     0738     0113
         MAINT       501     3375     0739     0742     0004
         MAINT       394     3375     0743     0946     0204
         MAINT       124     3375     0000     0958     0959 **OVERLAP**
         SYSDUMP1    124     3375     0000     0958     0959 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3375     0000     0000     0001
         MAINT       19D     3375     0001     0063     0063
         MAINT       201     3375     0064     0098     0035
         MAINT       292     3375     0099     0129     0031
         MAINT       392     3375     0130     0160     0031
         MAINT       492     3375     0161     0176     0016
         MAINT       692     3375     0177     0192     0016
         MAINT       594     3375     0193     0310     0118
         MAINT       423     3375     0311     0326     0016
         MAINT       593     3375     0327     0467     0141
         MAINT       495     3375     0468     0475     0008
         MAINT       592     3375     0476     0491     0016
         MAINT       892     3375     0492     0499     0008
         MAINT       491     3375     0500     0512     0013
         MAINT       791     3375     0513     0531     0019
         MAINT       293     3375     0532     0751     0220
         MAINT       294     3375     0752     0939     0188
         MAINT       125     3375     0000     0958     0959 **OVERLAP**
         SYSDUMP1    125     3375     0000     0958     0959 **OVERLAP**
```

Figure 57 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3375

VOLUME	USERID	CUU	DEVTYPE	START	END	SIZE	
XASRES	$ALLOC$	A01	3375	0000	0000	0001	
	$DIRECT$	A04	3375	0001	0004	0004	
	$CP-NUC$	A09	3375	0005	0020	0016	
	$SYSCKP$	A06	3375	0021	0024	0004	
	$SYSWRM$	A07	3375	0025	0028	0004	
	$PAGE$	A03	3375	0029	0066	0038	
	$SPOOL$	B01	3375	0067	0158	0092	
	MAINT	391	3375	0159	0182	0024	
	MAINT	596	3375	0183	0214	0032	
				215	222	8	GAP
	MAINT	190	3375	0223	0335	0113	
	MAINT	291	3375	0336	0359	0024	
	MAINT	49D	3375	0360	0422	0063	
	MAINT	5E5	3375	0423	0427	0005	
				428	440	13	GAP
	GCS	191	3375	0441	0446	0006	
	MAINT	395	3375	0447	0462	0016	
	MAINT	59E	3375	0463	0478	0016	
				479	486	8	GAP
	LGLOPR	191	3375	0487	0487	0001	
	CMSBATCH	195	3375	0488	0491	0004	
	MAINT	191	3375	0492	0538	0047	
				539	560	22	GAP
	$T-DISK$	B01	3375	0561	0591	0031	
	MAINT	89E	3375	0592	0607	0016	
	MAINT	896	3375	0608	0623	0016	
	MAINT	895	3375	0624	0639	0016	
	MAINT	595	3375	0640	0659	0020	
				660	675	16	GAP
	MAINT	295	3375	0676	0695	0020	
	OPERATOR	191	3375	0696	0703	0008	
	OP1	191	3375	0704	0705	0002	
	OPERATNS	191	3375	0706	0731	0026	
	AUTOLOG1	191	3375	0732	0733	0002	
	DISKACNT	191	3375	0734	0735	0002	
	IBMPSR	191	3375	0736	0737	0002	
	EREP	191	3375	0738	0741	0004	
	MAINT	193	3375	0742	0817	0076	
	MAINT	194	3375	0818	0899	0082	
	MAINT	123	3375	0000	0958	0959	**OVERLAP**
	SYSDUMP1	123	3375	0000	0958	0959	**OVERLAP**

Figure 57 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3375

Sample Files for a 3380 Starter System

IBM provides the following sample files with the VM/XA SP 3380 Starter System:

- HCPSYS ASSEMBLE for Starter System
- System directories for 3380, 3380-E4, and 3380-K
- System residence DASD allocation
- Minidisk maps of the system residence device.

Sample HCPSYS ASSEMBLE for a 3380

This is only a sample file. You need to tailor it to your installation before it can be used.

```
SYS TITLE 'HCPSYS80 - HCPSYS FOR VM/XA INSTALL BUILD'
************************************************************************
* 5664-308 (C) COPYRIGHT IBM CORPORATION - 1988,1989                   *
* LICENSED MATERIALS - PROPERTY OF IBM                                 *
* SEE COPYRIGHT INSTRUCTIONS, G120-2083                                *
************************************************************************
*****************************************************************
*
* MODULE NAME - HCPSYS
*
* DESCRIPTIVE NAME - DEFINITION OF SYSTEM
*
*
* FUNCTION - TO DEFINE THE SYSTEM
*
*
*   REGISTER CONVENTIONS - SYMBOLIC REFERENCES TO REGISTERS ARE
*                          OF THE FORM "RX" WHERE X IS A NUMBER
*                          RANGING FROM 0 TO 15 (SEE HCPEQUAT COPY)
*
*   PATCH LABEL - NONE
*
* MODULE TYPE - DATA
*
*   PROCESSOR - ASSEMBLER H (VERSION 2)
*
*   ATTRIBUTES - RESIDENT, DATA-ONLY
*
* MACROS -
*
*          SYSSTORE - DEFINE SYSTEM REAL STORAGE
*          SYSRES   - DEFINE SYSTEM RESIDENCE VOLUME
*          SYSCPVOL - DEFINE SYSTEM VOLUMES
*          SYSOPR   - DEFINE SYSTEM OPERATOR
*          SYSTIME  - DEFINE THE TIME ZONE
*          SYSACNT  - DEFINE ACCOUNTING USERID
*          SYSDUMP  - DEFINE DUMP RECEIVER USERID
*          SYSEREP  - DEFINE EREP RECEIVER USERID
*          SYSID    - DEFINE SYSTEM ID'S BY PROCESSOR
*          SYSUVOL  - DEFINE USER VOLUMES TO BE MOUNTED
*          SYSFCN   - DEFINE PRIV CLASSES ASSIGNED TO CP FUNCTIONS
*          SYSEND   - END SYSTEM GENERATION
*
*
* GENERAL COMMENTS - THIS MODULE IS THE USER'S RESPONSIBILITY.
```

```
*
*************************************************************************
         SPACE 1
*-------------------------------------------------------------
* THE SYSTEM DEFINITION MACROS SYSSTORE, SYSRES, SYSCPVOL,
* SYSOPR, AND SYSTIME, ARE CODED HERE, IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSSTORE RMSIZE=32M,VRSIZE=8M,VRFREE=256
         SYSRES SYSVOL=XASRES,                                        *
               SYSRES=123,SYSTYPE=3380,                               *
               SYSNUC=003,                                            *
               SYSCKP=(013,2),                                        *
               SYSWRM=(015,2)
         SYSTIME ZONE=5,LOC=WEST,ID=EST
         SYSOPR SYSOPER=OPERATOR
         SYSCPVOL     XASRES,XASERV,XAP001,                           *
               DRUM01,DRUM02,DRUM03,DRUM04,DRUM05,DRUM06,DRUM07,      *
               PAG001,PAG002,PAG003,PAG004,PAG005,PAG006,PAG007,      *
               SPL001,SPL002,SPL003,SPL004,SPL005,SPL006,SPL007
         SPACE 1
*-------------------------------------------------------------
* ANY OF THE OPTIONAL MACROS THAT ARE DESIRED ARE CODED HERE,
* IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSID      DEFAULT=INSTTST,(3081A,3081,10000)
         SYSACNT USERID=DISKACNT
         SYSDUMP USERID=OPERATNS
         SYSEREP USERID=EREP
         SYSUVOL      USRP01,USRP02,USRP03,USRP04,USRP05,USRP06,USRP07,*
               USRP08,USRP09
         SYSFCN  OPER=A,CPRD=CE,CPWT=C,SERV=F,DFLT=G
         SPACE 1
*-------------------------------------------------------------
* THE SYSEND MACRO IS CODED HERE
*-------------------------------------------------------------
         SPACE 1
         SYSEND
```

There are three sets of sample directory, minidisk allocations, and minidisk maps for different models of 3380 DASD. The first set ("Sample Directory for a 3380," "System Residence DASD Allocation for a 3380" on page 790 , and "Minidisk Maps for a 3380 System Residence Device" on page 791) is for the 3380. The second set ("Sample Directory for a 3380-E4" on page 793, "System Residence DASD Allocation for a 3380-E4" on page 804, and "Minidisk Maps for a 3380-E4 System Residence Device" on page 805) is for the 3380-E4. The third set ("Sample Directory for a 3380-K" on page 807, "System Residence DASD Allocation for a 3380-K" on page 818, and "Minidisk Maps for a 3380-K System Residence Device" on page 819) is for the 3380-K.

Sample Directory for a 3380

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1988, 1989 *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM    *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083         *
*************************************************************
*************************************************************
*  3380 SYSTEM   VM/XA SYSTEM  PRODUCT  DIRECTORY           *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.   *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,   *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.   *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH   *
*  YOUR HARDWARE ADDRESSES.                                 *
*                                                           *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                 *
*                                                           *
*      USER ANYONE SOMEPASS 3M 8M FG                        *
*       AUTOLOG AUTOLOG1 OP1 MAINT                          *
*        ACCOUNT ??????                                     *
*        IPL 190                                            *
*        CONSOLE 01F 3215                                   *
*        SPOOL 00C 2540 READER A                            *
*        SPOOL 00D 2540 PUNCH A                             *
*        SPOOL 00E 1403 A                                   *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS *
*                                                           *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN *
*                READ ONLY MODE.                            *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN *
*                WRITE MODE.                                *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN *
*                MULTI-WRITE MODE.                          *
*                                                           *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.   *
*                                                           *
*************************************************************
*                                                           *
*                                                           *
*  Following is a sample user entry and a brief description *
*  of each entry.  A more detailed description may be found *
*  by referencing the VM/XA System Product Planning and     *
```

```
*  Administration manual.                                        *
*                                                                *
*    USER CMS1  abc123  3M  32M G                                *
*      - Userid is CMS1                                          *
*      - Logon password is abc123                                *
*      - At logon time, virtual storage will equal 3M            *
*      - Maximum storage that may be defined equals 32M          *
*      - Privilege class is G                                    *
*                                                                *
*     AUTOLOG AUTOLOG1 OP1 MAINT                                 *
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,          *
*         OP1, and MAINT                                         *
*                                                                *
*     ACCOUNT ACT4 CMSTST                                        *
*       - Time used by CMS1 will be charged to account ACT4      *
*       - Printed output will be sent to distribution CMSTST     *
*                                                                *
*     IPL CMS                                                    *
*       - Will cause automatic IPL of named save system of       *
*         CMS                                                    *
*                                                                *
*     CONSOLE 009 3215                                           *
*       - Defines virtual machines console as a 3215 at a        *
*         virtual address of 009                                 *
*                                                                *
*     SPOOL 00C 2540 READER A                                    *
*     SPOOL 00D 2540 PUNCH A                                     *
*     SPOOL 00E 1403 A                                           *
*       - Defines virtual unit record devices of reader,         *
*         punch, and printer with a spooling class of 'A'        *
*                                                                *
*     LINK MAINT 190 190 RR                                      *
*       - Allows read access to minidisk 190 of user 'MAINT'     *
*                                                                *
*     LINK MAINT 19E 19E RR                                      *
*       - Allows read access to minidisk 19E of user 'MAINT'     *
*                                                                *
*     LINK MAINT 19D 19D RR                                      *
*       - Allows read access to minidisk 19d of user 'MAINT'     *
*                                                                *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS               *
*       - Defines minidisk with a virtual address of 999         *
*       - Specifies that the minidisk resides on a real 3350     *
*       - Minidisk starts at cylinder 236                        *
*       - Mdisk is 22 cylinders in size (starting at cyl 236     *
*       - The real volume serial number is 'CPPACK'              *
*       - Mdisk is to be accessed in write mode if no other      *
*         user has write access. Alternate access read-only      *
*       - RPASS is the required password for another user to     *
*         link to this minidisk in read mode                     *
*       - WPASS is the required password for another user to     *
*         link to this minidisk in write mode                    *
****************************************************************
*
*
*
DIRECTORY 123 3380 XASRES
*
****************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS              *
```

```
*****************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3380 000 001 XASRES R
 MDISK B01 3380 000 001 XASERV R
 MDISK C01 3380 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
 MDISK A04 3380 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3380 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3380 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3380 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3380 017 024 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3380 041 087 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3380 539 020 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                 PROP LOGICAL OPERATOR                             *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3380 538 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
```

```
*************************************************************
*             SYSTEM RELATED USERIDS                        *
*************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
 NAMESAVE GCS
 NAMESAVE VTAM
 NAMESAVE HELP
 NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3380 000 885 XASRES  MW
 MDISK 124 3380 000 885 XAP001  RR
 MDISK 125 3380 000 885 XASERV  RR

 MDISK 191 3380 694 030 XASRES  MW
 MDISK 193 3380 128 046 XASRES  MW
 MDISK 194 3380 416 056 XASRES  MW
 MDISK 201 3380 001 022 XASERV  MW RMAINT   WMAINT  MMAINT

 MDISK 292 3380 023 020 XASERV  MW READ WRITE MULTIPLE
 MDISK 392 3380 043 020 XASERV  MW READ WRITE MULTIPLE
 MDISK 492 3380 063 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 692 3380 073 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 594 3380 083 075 XASERV  MW READ WRITE MULTIPLE
 MDISK 593 3380 173 090 XASERV  MW READ WRITE MULTIPLE

 MDISK 295 3380 559 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3380 572 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3380 263 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3380 268 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3380 278 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 491 3380 283 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 791 3380 291 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 89E 3380 526 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3380 303 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 895 3380 313 010 XASERV  MW READ WRITE MULTIPLE

 MDISK 490 3380 323 072 XASERV  MW
 MDISK 423 3380 516 010 XASRES  MW
 MDISK 293 3380 395 140 XASERV  MW RCMSAUX  WCMSAUX MCMSAUX
 MDISK 294 3380 535 120 XASERV  MW RCPAUX   WCPAUX  MCPAUX
 MDISK 393 3380 752 066 XASRES  WR
 MDISK 394 3380 655 130 XASERV  WR
 MDISK 19C 3380 194 040 XASRES  WR
 MDISK 49C 3380 833 040 XASRES  WR

 MDISK 291 3380 818 015 XASRES  MW
 MDISK 391 3380 158 015 XASERV  MW
 MDISK 591 3380 785 095 XASERV  MW READ WRITE MULTIPLE
 MDISK 691 3380 306 110 XASRES  MW READ WRITE MULTIPLE
 MDISK 49D 3380 582 040 XASRES  MW READ WRITE MULTIPLE
 MDISK 501 3380 646 002 XASRES  MW READ WRITE MULTIPLE
 MDISK 595 3380 622 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 5E5 3380 873 005 XASRES  MW READ WRITE MULTIPLE
```

```
 MDISK 190 3380 234 072 XASRES  MW ALL
 MDISK 19E 3380 650 044 XASRES  MW ALL
 MDISK 19D 3380 476 040 XASRES  MW ALL
 MDISK 596 3380 174 020 XASRES  MW ALL
 MDISK 59E 3380 634 010 XASRES  MW ALL
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3380 644 002 XASRES RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3380 536 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3380 472 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT NOLOG    3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
  IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
```

```
 LINK MAINT 19D 19D RR
 MDISK 191 3380 724 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 729 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3380 000 885 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
```

```
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
  MDISK 191 3380 750 002 XASRES WR READ     WRITE   MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 730 017 XASRES MR RDVF     WDVF     MDVF
*
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3380 747 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 748 001 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
```

```
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 749 001 XASRES WR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3380 000 885 XASRES RR
 MDISK 124 3380 000 885 XAP001 RR
 MDISK 125 3380 000 885 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
```

```
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3380

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3380 000 885 XASRES  MW
 MDISK 124 3380 000 885 XAP001  RR
 MDISK 125 3380 000 885 XASERV  RR

 MDISK 191 3380 694 030 XASRES  MW
 MDISK 193 3380 128 046 XASRES  MW
 MDISK 194 3380 416 056 XASRES  MW
 MDISK 201 3380 001 022 XASERV  MW RMAINT   WMAINT  MMAINT

 MDISK 292 3380 023 020 XASERV  MW READ WRITE MULTIPLE
 MDISK 392 3380 043 020 XASERV  MW READ WRITE MULTIPLE
 MDISK 492 3380 063 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 692 3380 073 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 594 3380 083 075 XASERV  MW READ WRITE MULTIPLE
 MDISK 593 3380 173 090 XASERV  MW READ WRITE MULTIPLE

 MDISK 295 3380 559 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3380 572 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3380 263 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3380 268 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3380 278 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 491 3380 283 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 791 3380 291 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 89E 3380 526 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3380 303 010 XASERV  MW READ WRITE MULTIPLE
 MDISK 895 3380 313 010 XASERV  MW READ WRITE MULTIPLE

 MDISK 490 3380 323 072 XASERV  MW
 MDISK 423 3380 516 010 XASRES  MW
 MDISK 293 3380 395 140 XASERV  MW RCMSAUX  WCMSAUX MCMSAUX
 MDISK 294 3380 535 120 XASERV  MW RCPAUX   WCPAUX  MCPAUX
 MDISK 393 3380 752 066 XASRES  WR
 MDISK 394 3380 655 130 XASERV  WR
 MDISK 19C 3380 194 040 XASRES  WR
 MDISK 49C 3380 833 040 XASRES  WR

 MDISK 291 3380 818 015 XASRES  MW
 MDISK 391 3380 158 015 XASERV  MW
 MDISK 591 3380 785 095 XASERV  MW READ WRITE MULTIPLE
 MDISK 691 3380 306 110 XASRES  MW READ WRITE MULTIPLE
 MDISK 49D 3380 582 040 XASRES  MW READ WRITE MULTIPLE
```

```
  MDISK 501 3380 646 002 XASRES  MW READ WRITE MULTIPLE
  MDISK 595 3380 622 012 XASRES  MW READ WRITE MULTIPLE
  MDISK 5E5 3380 873 005 XASRES  MW READ WRITE MULTIPLE

  MDISK 190 3380 234 072 XASRES  MW ALL
  MDISK 19E 3380 650 044 XASRES  MW ALL
  MDISK 19D 3380 476 040 XASRES  MW ALL
  MDISK 596 3380 174 020 XASRES  MW ALL
  MDISK 59E 3380 634 010 XASRES  MW ALL
 *
 * MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
  MDISK 36E 3380 644 002 XASRES RR RPVM     WPVM     MPVM
  LINK RSCS  191 499  MW
  LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3380

Figure 58 shows the allocation for the system residence device for the 3380 VM/XA SP starter system. The allocation and minidisk maps for the 3380-E4 ("System Residence DASD Allocation for a 3380-E4" on page 804 and "Minidisk Maps for a 3380-E4 System Residence Device" on page 805) and 3380-K ("System Residence DASD Allocation for a 3380-K" on page 818 and "Minidisk Maps for a 3380-K System Residence Device" on page 819) follow the corresponding sample directories ("Sample Directory for a 3380-E4" on page 793 and "Sample Directory for a 3380-K" on page 807, respectively).

```
TYPE      CYL  CYL
PERM      0000-0000
DRCT      0001-0002
PERM      0003-0016
PAGE      0017-0040
SPOL      0041-0127
PERM      0128-0538
TDSK      0539-0558
PERM      0559-0884
```

Figure 58. System Residence DASD Allocation for a 3380

Minidisk Maps for a 3380 System Residence Device

Figure 59 shows the minidisk maps of the system residence device for the 3380. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3380     0000     0884     0885

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3380     0000     0000     0001
         MAINT       124     3380     0000     0884     0885 **OVERLAP**
         SYSDUMP1    124     3380     0000     0884     0885 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3380     0000     0000     0001
         MAINT       201     3380     0001     0022     0022
         MAINT       292     3380     0023     0042     0020
         MAINT       392     3380     0043     0062     0020
         MAINT       492     3380     0063     0072     0010
         MAINT       692     3380     0073     0082     0010
         MAINT       594     3380     0083     0157     0075
         MAINT       391     3380     0158     0172     0015
         MAINT       593     3380     0173     0262     0090
         MAINT       495     3380     0263     0267     0005
         MAINT       592     3380     0268     0277     0010
         MAINT       892     3380     0278     0282     0005
         MAINT       491     3380     0283     0290     0008
         MAINT       791     3380     0291     0302     0012
         MAINT       896     3380     0303     0312     0010
         MAINT       895     3380     0313     0322     0010
         MAINT       490     3380     0323     0394     0072
         MAINT       293     3380     0395     0534     0140
         MAINT       294     3380     0535     0654     0120
         MAINT       394     3380     0655     0784     0130
         MAINT       591     3380     0785     0879     0095
         MAINT       125     3380     0000     0884     0885 **OVERLAP**
         SYSDUMP1    125     3380     0000     0884     0885 **OVERLAP**
```

Figure 59 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3380

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END      SIZE
XASRES   $ALLOC$     A01     3380     0000      0000      0001
         $DIRECT$    A04     3380     0001      0002      0002
         $CP-NUC$    A09     3380     0003      0012      0010
         $SYSCKP$    A06     3380     0013      0014      0002
         $SYSWRM$    A07     3380     0015      0016      0002
         $PAGE$      A03     3380     0017      0040      0024
         $SPOOL$     B01     3380     0041      0127      0087
         MAINT       193     3380     0128      0173      0046
         MAINT       596     3380     0174      0193      0020
         MAINT       19C     3380     0194      0233      0040
         MAINT       190     3380     0234      0305      0072
         MAINT       691     3380     0306      0415      0110
         MAINT       194     3380     0416      0471      0056
         GCS         191     3380     0472      0475      0004
         MAINT       19D     3380     0476      0515      0040
         MAINT       423     3380     0516      0525      0010
         MAINT       89E     3380     0526      0535      0010
         CMSBATCH    195     3380     0536      0537      0002
         LGLOPR      191     3380     0538      0538      0001
         $T-DISK$    B01     3380     0539      0558      0020
         MAINT       295     3380     0559      0571      0013
         MAINT       395     3380     0572      0581      0010
         MAINT       49D     3380     0582      0621      0040
         MAINT       595     3380     0622      0633      0012
         MAINT       59E     3380     0634      0643      0010
         MAINT       36E     3380     0644      0645      0002
         MAINT       501     3380     0646      0647      0002
                                       648       649         2    GAP
         MAINT       19E     3380     0650      0693      0044
         MAINT       191     3380     0694      0723      0030
         OPERATOR    191     3380     0724      0728      0005
         OP1         191     3380     0729      0729      0001
         OPERATNS    191     3380     0730      0746      0017
         AUTOLOG1    191     3380     0747      0747      0001
         DISKACNT    191     3380     0748      0748      0001
         IBMPSR      191     3380     0749      0749      0001
         EREP        191     3380     0750      0751      0002
         MAINT       393     3380     0752      0817      0066
         MAINT       291     3380     0818      0832      0015
         MAINT       49C     3380     0833      0872      0040
         MAINT       5E5     3380     0873      0877      0005
         MAINT       123     3380     0000      0884      0885 **OVERLAP**
         SYSDUMP1    123     3380     0000      0884      0885 **OVERLAP**
```

Figure 59 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3380

Sample Directory for a 3380-E4

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1988, 1989 *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM    *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083         *
*************************************************************
*************************************************************
*  3380-E4 SYSTEM VM/XA SYSTEM  PRODUCT  DIRECTORY          *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.   *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST  *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,     *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.     *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL   *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH     *
*  YOUR HARDWARE ADDRESSES.                                   *
*                                                             *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO  *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE   *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY   *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                   *
*                                                             *
*      USER ANYONE SOMEPASS 3M 8M FG                          *
*       AUTOLOG AUTOLOG1 OP1 MAINT                            *
*        ACCOUNT ??????                                       *
*        IPL 190                                              *
*        CONSOLE 01F 3215                                     *
*        SPOOL 00C 2540 READER A                              *
*        SPOOL 00D 2540 PUNCH A                               *
*        SPOOL 00E 1403 A                                     *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS   *
*                                                             *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                READ ONLY MODE.                              *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                WRITE MODE.                                  *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                MULTI-WRITE MODE.                            *
*                                                             *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.     *
*                                                             *
*************************************************************
*                                                             *
*                                                             *
*  Following is a sample user entry and a brief description   *
*  of each entry.  A more detailed description may be found   *
*  by referencing the VM/XA System Product Planning and       *
*  Administration manual.                                     *
*                                                             *
*    USER CMS1  abc123  3M  32M G                             *
*       - Userid is CMS1                                      *
*       - Logon password is abc123                            *
*       - At logon time, virtual storage will equal 3M        *
*       - Maximum storage that may be defined equals 32M      *
*       - Privilege class is G                                *
*                                                             *
*     AUTOLOG AUTOLOG1 OP1 MAINT                              *
```

```
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,      *
*         OP1, and MAINT                                     *
*                                                            *
*     ACCOUNT ACT4 CMSTST                                    *
*       - Time used by CMS1 will be charged to account ACT4  *
*       - Printed output will be sent to distribution CMSTST *
*                                                            *
*     IPL CMS                                                *
*       - Will cause automatic IPL of named save system of   *
*         CMS                                                *
*                                                            *
*     CONSOLE 009 3215                                       *
*       - Defines virtual machines console as a 3215 at a    *
*         virtual address of 009                             *
*                                                            *
*     SPOOL 00C 2540 READER A                                *
*     SPOOL 00D 2540 PUNCH A                                 *
*     SPOOL 00E 1403 A                                       *
*       - Defines virtual unit record devices of reader,     *
*         punch, and printer with a spooling class of 'A'    *
*                                                            *
*     LINK MAINT 190 190 RR                                  *
*       - Allows read access to minidisk 190 of user 'MAINT' *
*                                                            *
*     LINK MAINT 19E 19E RR                                  *
*       - Allows read access to minidisk 19E of user 'MAINT' *
*                                                            *
*     LINK MAINT 19D 19D RR                                  *
*       - Allows read access to minidisk 19d of user 'MAINT' *
*                                                            *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS           *
*       - Defines minidisk with a virtual address of 999     *
*       - Specifies that the minidisk resides on a real 3350 *
*       - Minidisk starts at cylinder 236                    *
*       - Mdisk is 22 cylinders in size (starting at cyl 236 *
*       - The real volume serial number is 'CPPACK'          *
*       - Mdisk is to be accessed in write mode if no other  *
*         user has write access. Alternate access read-only  *
*       - RPASS is the required password for another user to *
*         link to this minidisk in read mode                 *
*       - WPASS is the required password for another user to *
*         link to this minidisk in write mode                *
*************************************************************
*
*
*
DIRECTORY 123 3380 XASRES
*
*************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS           *
*************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3380 000 001 XASRES R
 MDISK B01 3380 000 001 XASERV R
 MDISK C01 3380 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
```

```
 MDISK A04 3380 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3380 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3380 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3380 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3380 017 024 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3380 041 087 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3380 539 020 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                   PROP LOGICAL OPERATOR                           *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3380 233 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
```

```
NAMESAVE GCS
NAMESAVE VTAM
NAMESAVE HELP
NAMESAVE INSTHELP
CONSOLE 009 3215 T
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
MDISK 123 3380 0000 1770 XASRES  MW
MDISK 124 3380 0000 1770 XAP001  RR
MDISK 125 3380 0000 1770 XASERV  RR

MDISK 191 3380 0694 030 XASRES  MW
MDISK 193 3380 0128 046 XASRES  MW
MDISK 194 3380 0362 056 XASRES  MW
MDISK 201 3380 1711 022 XASRES  MW RMAINT   WMAINT   MMAINT

MDISK 292 3380 1743 020 XASRES  MW READ WRITE MULTIPLE
MDISK 392 3380 1453 020 XASRES  MW READ WRITE MULTIPLE
MDISK 492 3380 1473 010 XASRES  MW READ WRITE MULTIPLE
MDISK 692 3380 1483 010 XASRES  MW READ WRITE MULTIPLE

MDISK 593 3380 1082 090 XASRES  MW READ WRITE MULTIPLE
MDISK 594 3380 1007 075 XASRES  MW READ WRITE MULTIPLE

MDISK 295 3380 1493 013 XASRES  MW READ WRITE MULTIPLE
MDISK 395 3380 1506 010 XASRES  MW READ WRITE MULTIPLE
MDISK 495 3380 0725 005 XASRES  MW READ WRITE MULTIPLE
MDISK 592 3380 1523 010 XASRES  MW READ WRITE MULTIPLE
MDISK 892 3380 1518 005 XASRES  MW READ WRITE MULTIPLE
MDISK 491 3380 0527 008 XASRES  MW READ WRITE MULTIPLE
MDISK 791 3380 1436 012 XASRES  MW READ WRITE MULTIPLE
MDISK 89E 3380 1533 010 XASRES  MW READ WRITE MULTIPLE
MDISK 5E5 3380 0458 005 XASRES  MW READ WRITE MULTIPLE
MDISK 896 3380 1543 010 XASRES  MW READ WRITE MULTIPLE
MDISK 895 3380 1553 010 XASRES  MW READ WRITE MULTIPLE

MDISK 490 3380 1573 072 XASRES  MW
MDISK 423 3380 1563 010 XASRES  MW
MDISK 293 3380 0747 140 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
MDISK 294 3380 0887 120 XASRES  MW RCPAUX   WCPAUX   MCPAUX
MDISK 393 3380 1645 066 XASRES  WR
MDISK 394 3380 0559 130 XASRES  WR
MDISK 19C 3380 1396 040 XASRES  WR
MDISK 49C 3380 0467 040 XASRES  WR
MDISK 596 3380 0507 020 XASRES  WR
MDISK 59E 3380 1733 010 XASRES  WR

MDISK 391 3380 0218 015 XASRES  MW
MDISK 591 3380 1172 095 XASRES  MW READ WRITE MULTIPLE
MDISK 691 3380 1267 110 XASRES  MW READ WRITE MULTIPLE
MDISK 49D 3380 0306 040 XASRES  MW READ WRITE MULTIPLE
MDISK 501 3380 1394 002 XASRES  MW READ WRITE MULTIPLE
MDISK 595 3380 0346 012 XASRES  MW READ WRITE MULTIPLE
MDISK 291 3380 1379 015 XASRES  MW

MDISK 190 3380 0234 072 XASRES  MW ALL
MDISK 19E 3380 0174 044 XASRES  MW ALL
```

```
 MDISK 19D 3380 0418 040 XASRES  MW ALL
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3380 0537 002 XASRES RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3380 1377 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3380 463 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT NOLOG    3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
  IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 689 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
```

```
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 724 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3380 000 1770 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
```

```
  MDISK 191 3380 1448 002 XASRES WR READ     WRITE    MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 730 017 XASRES MR RDVF     WDVF     MDVF
*
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3380 535 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 536 001 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
```

```
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 1450 001 XASRES WR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3380 000 1770 XASRES RR
 MDISK 124 3380 000 1770 XAP001 RR
 MDISK 125 3380 000 1770 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
```

```
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3380-E4

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3380 0000 1770 XASRES  MW
 MDISK 124 3380 0000 1770 XAP001  RR
 MDISK 125 3380 0000 1770 XASERV  RR

 MDISK 191 3380 0694 030 XASRES  MW
 MDISK 193 3380 0128 046 XASRES  MW
 MDISK 194 3380 0362 056 XASRES  MW
 MDISK 201 3380 1711 022 XASRES  MW RMAINT   WMAINT   MMAINT

 MDISK 292 3380 1743 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 392 3380 1453 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 492 3380 1473 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 692 3380 1483 010 XASRES  MW READ WRITE MULTIPLE

 MDISK 593 3380 1082 090 XASRES  MW READ WRITE MULTIPLE
 MDISK 594 3380 1007 075 XASRES  MW READ WRITE MULTIPLE

 MDISK 295 3380 1493 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 395 3380 1506 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3380 0725 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 592 3380 1523 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 892 3380 1518 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 491 3380 0527 008 XASRES  MW READ WRITE MULTIPLE
 MDISK 791 3380 1436 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 89E 3380 1533 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 5E5 3380 0458 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3380 1543 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 895 3380 1553 010 XASRES  MW READ WRITE MULTIPLE

 MDISK 490 3380 1573 072 XASRES  MW
 MDISK 423 3380 1563 010 XASRES  MW
 MDISK 293 3380 0747 140 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3380 0887 120 XASRES  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 393 3380 1645 066 XASRES  WR
 MDISK 394 3380 0559 130 XASRES  WR
 MDISK 19C 3380 1396 040 XASRES  WR
 MDISK 49C 3380 0467 040 XASRES  WR
 MDISK 596 3380 0507 020 XASRES  WR
 MDISK 59E 3380 1733 010 XASRES  WR

 MDISK 391 3380 0218 015 XASRES  MW
```

```
  MDISK 591 3380 1172 095 XASRES  MW READ WRITE MULTIPLE
  MDISK 691 3380 1267 110 XASRES  MW READ WRITE MULTIPLE
  MDISK 49D 3380 0306 040 XASRES  MW READ WRITE MULTIPLE
  MDISK 501 3380 1394 002 XASRES  MW READ WRITE MULTIPLE
  MDISK 595 3380 0346 012 XASRES  MW READ WRITE MULTIPLE
  MDISK 291 3380 1379 015 XASRES  MW

  MDISK 190 3380 0234 072 XASRES  MW ALL
  MDISK 19E 3380 0174 044 XASRES  MW ALL
  MDISK 19D 3380 0418 040 XASRES  MW ALL
 *
 * MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
  MDISK 36E 3380 0537 002 XASRES RR RPVM     WPVM     MPVM
  LINK RSCS  191 499  MW
  LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3380-E4

Figure 60 shows the allocation for the system residence device for the 3380-E4 VM/XA SP Starter System.

```
TYPE  CYL  CYL
PERM 0000-0000
DRCT 0001-0002
PERM 0003-0016
PAGE 0017-0040
SPOL 0041-0127
PERM 0128-0538
TDSK 0539-0558
PERM 0559-1769
```

Figure 60. System Residence DASD Allocation for a 3380-E4

Minidisk Maps for a 3380-E4 System Residence Device

Figure 61 shows the minidisk maps of the system residence device for the 3380-E4. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3380     0000     1769     1770

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3380     0000     0000     0001
         MAINT       124     3380     0000     1769     1770 **OVERLAP**
         SYSDUMP1    124     3380     0000     1769     1770 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3380     0000     0000     0001
         MAINT       125     3380     0000     1769     1770 **OVERLAP**
         SYSDUMP1    125     3380     0000     1769     1770 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASRES   $ALLOC$     A01     3380     0000     0000     0001
         $DIRECT$    A04     3380     0001     0002     0002
         $CP-NUC$    A09     3380     0003     0012     0010
         $SYSCKP$    A06     3380     0013     0014     0002
         $SYSWRM$    A07     3380     0015     0016     0002
         $PAGE$      A03     3380     0017     0040     0024
         $SPOOL$     B01     3380     0041     0127     0087
         MAINT       193     3380     0128     0173     0046
         MAINT       19E     3380     0174     0217     0044
         MAINT       391     3380     0218     0232     0015
         LGLOPR      191     3380     0233     0233     0001
         MAINT       190     3380     0234     0305     0072
         MAINT       49D     3380     0306     0345     0040
         MAINT       595     3380     0346     0357     0012
                                      358      361        4    GAP
         MAINT       194     3380     0362     0417     0056
         MAINT       19D     3380     0418     0457     0040
         MAINT       5E5     3380     0458     0462     0005
         GCS         191     3380     0463     0466     0004
         MAINT       49C     3380     0467     0506     0040
         MAINT       596     3380     0507     0526     0020
         MAINT       491     3380     0527     0534     0008
         AUTOLOG1    191     3380     0535     0535     0001
```

Figure 61 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3380-E4

DISKACNT	191	3380	0536	0536	0001	
MAINT	36E	3380	0537	0538	0002	
$T-DISK$	B01	3380	0539	0558	0020	
MAINT	394	3380	0559	0688	0130	
OPERATOR	191	3380	0689	0693	0005	
MAINT	191	3380	0694	0723	0030	
OP1	191	3380	0724	0724	0001	
MAINT	495	3380	0725	0729	0005	
OPERATNS	191	3380	0730	0746	0017	
MAINT	293	3380	0747	0886	0140	
MAINT	294	3380	0887	1006	0120	
MAINT	594	3380	1007	1081	0075	
MAINT	593	3380	1082	1171	0090	
MAINT	591	3380	1172	1266	0095	
MAINT	691	3380	1267	1376	0110	
CMSBATCH	195	3380	1377	1378	0002	
MAINT	291	3380	1379	1393	0015	
MAINT	501	3380	1394	1395	0002	
MAINT	19C	3380	1396	1435	0040	
MAINT	791	3380	1436	1447	0012	
EREP	191	3380	1448	1449	0002	
IBMPSR	191	3380	1450	1450	0001	
			1451	1452	2	GAP
MAINT	392	3380	1453	1472	0020	
MAINT	492	3380	1473	1482	0010	
MAINT	692	3380	1483	1492	0010	
MAINT	295	3380	1493	1505	0013	
MAINT	395	3380	1506	1515	0010	
			1516	1517	2	GAP
MAINT	892	3380	1518	1522	0005	
MAINT	592	3380	1523	1532	0010	
MAINT	89E	3380	1533	1542	0010	
MAINT	896	3380	1543	1552	0010	
MAINT	895	3380	1553	1562	0010	
MAINT	423	33β0	1563	1572	0010	
MAINT	490	3380	1573	1644	0072	
MAINT	393	3380	1645	1710	0066	
MAINT	201	3380	1711	1732	0022	
MAINT	59E	3380	1733	1742	0010	
MAINT	292	3380	1743	1762	0020	
MAINT	123	3380	0000	1769	1770	**OVERLAP**
SYSDUMP1	123	3380	0000	1769	1770	**OVERLAP**

Figure 61 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3380-E4

Sample Directory for a 3380-K

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1988, 1989 *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM    *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083         *
*************************************************************
*************************************************************
*  3380-3 SYSTEM VM/XA SYSTEM  PRODUCT  DIRECTORY           *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.   *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST  *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,     *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.     *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL   *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH     *
*  YOUR HARDWARE ADDRESSES.                                   *
*                                                             *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO  *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE   *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY   *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                   *
*                                                             *
*      USER ANYONE SOMEPASS 3M 8M FG                          *
*       AUTOLOG AUTOLOG1 OP1 MAINT                            *
*        ACCOUNT ??????                                       *
*        IPL 190                                              *
*        CONSOLE 01F 3215                                     *
*        SPOOL 00C 2540 READER A                              *
*        SPOOL 00D 2540 PUNCH A                               *
*        SPOOL 00E 1403 A                                     *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS   *
*                                                             *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                READ ONLY MODE.                              *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                WRITE MODE.                                  *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN   *
*                MULTI-WRITE MODE.                            *
*                                                             *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.     *
*                                                             *
*************************************************************
*                                                            *
*                                                            *
*  Following is a sample user entry and a brief description  *
*  of each entry.  A more detailed description may be found  *
*  by referencing the VM/XA System Product Planning and      *
*  Administration manual.                                    *
*                                                            *
*    USER CMS1  abc123  3M  32M G                            *
*       - Userid is CMS1                                     *
*       - Logon password is abc123                           *
*       - At logon time, virtual storage will equal 3M       *
*       - Maximum storage that may be defined equals 32M     *
*       - Privilege class is G                               *
*                                                            *
*     AUTOLOG AUTOLOG1 OP1 MAINT                             *
```

```
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,      *
*         OP1, and MAINT                                     *
*                                                            *
*     ACCOUNT ACT4 CMSTST                                    *
*       - Time used by CMS1 will be charged to account ACT4  *
*       - Printed output will be sent to distribution CMSTST *
*                                                            *
*     IPL CMS                                                *
*       - Will cause automatic IPL of named save system of   *
*         CMS                                                *
*                                                            *
*     CONSOLE 009 3215                                       *
*       - Defines virtual machines console as a 3215 at a    *
*         virtual address of 009                             *
*                                                            *
*     SPOOL 00C 2540 READER A                                *
*     SPOOL 00D 2540 PUNCH A                                 *
*     SPOOL 00E 1403 A                                       *
*       - Defines virtual unit record devices of reader,     *
*         punch, and printer with a spooling class of 'A'    *
*                                                            *
*     LINK MAINT 190 190 RR                                  *
*       - Allows read access to minidisk 190 of user 'MAINT' *
*                                                            *
*     LINK MAINT 19E 19E RR                                  *
*       - Allows read access to minidisk 19E of user 'MAINT' *
*                                                            *
*     LINK MAINT 19D 19D RR                                  *
*       - Allows read access to minidisk 19d of user 'MAINT' *
*                                                            *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS           *
*       - Defines minidisk with a virtual address of 999     *
*       - Specifies that the minidisk resides on a real 3350 *
*       - Minidisk starts at cylinder 236                    *
*       - Mdisk is 22 cylinders in size (starting at cyl 236 *
*       - The real volume serial number is 'CPPACK'          *
*       - Mdisk is to be accessed in write mode if no other  *
*         user has write access. Alternate access read-only  *
*       - RPASS is the required password for another user to *
*         link to this minidisk in read mode                 *
*       - WPASS is the required password for another user to *
*         link to this minidisk in write mode                *
**************************************************************
*
*
*
DIRECTORY 123 3380 XASRES
*
**************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS            *
**************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3380 000 001 XASRES R
 MDISK B01 3380 000 001 XASERV R
 MDISK C01 3380 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
```

```
 MDISK A04 3380 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3380 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3380 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3380 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3380 017 024 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3380 041 087 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3380 539 020 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                  PROP LOGICAL OPERATOR                            *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3380 532 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
```

```
NAMESAVE GCS
NAMESAVE VTAM
NAMESAVE HELP
NAMESAVE INSTHELP
CONSOLE 009 3215 T
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
MDISK 123 3380 0000 2655 XASRES  MW
MDISK 124 3380 0000 2655 XAP001  RR
MDISK 125 3380 0000 2655 XASERV  RR

MDISK 191 3380 0694 030 XASRES  MW
MDISK 193 3380 0176 046 XASRES  MW
MDISK 194 3380 0336 056 XASRES  MW
MDISK 201 3380 0787 022 XASRES  MW RMAINT   WMAINT   MMAINT

MDISK 292 3380 0809 020 XASRES  MW READ WRITE MULTIPLE
MDISK 392 3380 0128 020 XASRES  MW READ WRITE MULTIPLE
MDISK 492 3380 0148 010 XASRES  MW READ WRITE MULTIPLE
MDISK 692 3380 0851 010 XASRES  MW READ WRITE MULTIPLE

MDISK 593 3380 0861 090 XASRES  MW READ WRITE MULTIPLE
MDISK 594 3380 0951 075 XASRES  MW READ WRITE MULTIPLE
MDISK 490 3380 1433 072 XASRES  MW
MDISK 423 3380 0166 010 XASRES  MW
MDISK 293 3380 0392 140 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
MDISK 294 3380 1093 120 XASRES  MW RCPAUX   WCPAUX   MCPAUX
MDISK 393 3380 1819 066 XASRES  WR
MDISK 394 3380 1213 130 XASRES  WR
MDISK 19C 3380 1343 040 XASRES  WR
MDISK 49C 3380 0747 040 XASRES  WR
MDISK 596 3380 1383 020 XASRES  WR
MDISK 59E 3380 1403 010 XASRES  WR
MDISK 5E5 3380 1572 005 XASRES  WR

MDISK 295 3380 1026 013 XASRES  WR READ WRITE MULTIPLE
MDISK 395 3380 0829 010 XASRES  WR READ WRITE MULTIPLE
MDISK 495 3380 1423 005 XASRES  WR READ WRITE MULTIPLE
MDISK 592 3380 0673 010 XASRES  WR READ WRITE MULTIPLE
MDISK 892 3380 1428 005 XASRES  WR READ WRITE MULTIPLE
MDISK 491 3380 0158 008 XASRES  WR READ WRITE MULTIPLE
MDISK 791 3380 0222 012 XASRES  WR READ WRITE MULTIPLE
MDISK 89E 3380 1530 010 XASRES  WR READ WRITE MULTIPLE
MDISK 896 3380 1540 010 XASRES  WR READ WRITE MULTIPLE
MDISK 895 3380 1550 010 XASRES  WR READ WRITE MULTIPLE

MDISK 391 3380 0306 015 XASRES  MW
MDISK 591 3380 1577 095 XASRES  MW READ WRITE MULTIPLE
MDISK 691 3380 0559 110 XASRES  MW READ WRITE MULTIPLE
MDISK 49D 3380 1885 040 XASRES  MW READ WRITE MULTIPLE
MDISK 501 3380 1672 002 XASRES  MW READ WRITE MULTIPLE
MDISK 291 3380 0321 015 XASRES  MW
MDISK 595 3380 1560 012 XASRES  MW READ WRITE MULTIPLE

MDISK 190 3380 0234 072 XASRES  MW ALL
MDISK 19E 3380 1674 044 XASRES  MW ALL
MDISK 19D 3380 1925 040 XASRES  MW ALL
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
```

```
 MDISK 36E 3380 0537 002 XASRES RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3380 533 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3380 669 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT NOLOG    3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
  IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 725 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
```

```
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 724 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3380 000 2655 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
  MDISK 191 3380 841 002 XASRES WR READ     WRITE   MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
```

```
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 730 017 XASRES MR RDVF     WDVF     MDVF
*
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3380 839 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3380 693 001 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
```

```
 MDISK 191 3380 840 001 XASRES WR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3380 000 2655 XASRES RR
 MDISK 124 3380 000 2655 XAP001 RR
 MDISK 125 3380 000 2655 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
```

```
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3380-K

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3380 0000 2655 XASRES  MW
 MDISK 124 3380 0000 2655 XAP001  RR
 MDISK 125 3380 0000 2655 XASERV  RR

 MDISK 191 3380 0694 030 XASRES  MW
 MDISK 193 3380 0176 046 XASRES  MW
 MDISK 194 3380 0336 056 XASRES  MW
 MDISK 201 3380 0787 022 XASRES  MW RMAINT   WMAINT   MMAINT

 MDISK 292 3380 0809 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 392 3380 0128 020 XASRES  MW READ WRITE MULTIPLE
 MDISK 492 3380 0148 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 692 3380 0851 010 XASRES  MW READ WRITE MULTIPLE

 MDISK 593 3380 0861 090 XASRES  MW READ WRITE MULTIPLE
 MDISK 594 3380 0951 075 XASRES  MW READ WRITE MULTIPLE
 MDISK 490 3380 1433 072 XASRES  MW
 MDISK 423 3380 0166 010 XASRES  MW
 MDISK 293 3380 0392 140 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3380 1093 120 XASRES  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 393 3380 1819 066 XASRES  WR
 MDISK 394 3380 1213 130 XASRES  WR
 MDISK 19C 3380 1343 040 XASRES  WR
 MDISK 49C 3380 0747 040 XASRES  WR
 MDISK 596 3380 1383 020 XASRES  WR
 MDISK 59E 3380 1403 010 XASRES  WR
 MDISK 5E5 3380 1572 005 XASRES  WR

 MDISK 295 3380 1026 013 XASRES  WR READ WRITE MULTIPLE
 MDISK 395 3380 0829 010 XASRES  WR READ WRITE MULTIPLE
 MDISK 495 3380 1423 005 XASRES  WR READ WRITE MULTIPLE
 MDISK 592 3380 0673 010 XASRES  WR READ WRITE MULTIPLE
 MDISK 892 3380 1428 005 XASRES  WR READ WRITE MULTIPLE
 MDISK 491 3380 0158 008 XASRES  WR READ WRITE MULTIPLE
 MDISK 791 3380 0222 012 XASRES  WR READ WRITE MULTIPLE
 MDISK 89E 3380 1530 010 XASRES  WR READ WRITE MULTIPLE
 MDISK 896 3380 1540 010 XASRES  WR READ WRITE MULTIPLE
 MDISK 895 3380 1550 010 XASRES  WR READ WRITE MULTIPLE

 MDISK 391 3380 0306 015 XASRES  MW
 MDISK 591 3380 1577 095 XASRES  MW READ WRITE MULTIPLE
```

```
  MDISK 691 3380 0559 110 XASRES  MW READ WRITE MULTIPLE
  MDISK 49D 3380 1885 040 XASRES  MW READ WRITE MULTIPLE
  MDISK 501 3380 1672 002 XASRES  MW READ WRITE MULTIPLE
  MDISK 291 3380 0321 015 XASRES  MW
  MDISK 595 3380 1560 012 XASRES  MW READ WRITE MULTIPLE

  MDISK 190 3380 0234 072 XASRES  MW ALL
  MDISK 19E 3380 1674 044 XASRES  MW ALL
  MDISK 19D 3380 1925 040 XASRES  MW ALL
 *
 * MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
  MDISK 36E 3380 0537 002 XASRES RR RPVM     WPVM     MPVM
  LINK RSCS  191 499  MW
  LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3380-K

Figure 62 shows the allocation for the system residence device for the 3380-K VM/XA SP Starter System.

```
TYPE  CYL  CYL
PERM 0000-0000
DRCT 0001-0002
PERM 0003-0016
PAGE 0017-0040
SPOL 0041-0127
PERM 0128-0538
TDSK 0539-0558
PERM 0559-2654
```

Figure 62. System Residence DASD Allocation for a 3380-K

Minidisk Maps for a 3380-K System Residence Device

Figure 63 shows the minidisk maps of the system residence device for the 3380-K. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3380     0000     2654     2655

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3380     0000     0000     0001
         MAINT       124     3380     0000     2654     2655 **OVERLAP**
         SYSDUMP1    124     3380     0000     2654     2655 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3380     0000     0000     0001
         MAINT       125     3380     0000     2654     2655 **OVERLAP**
         SYSDUMP1    125     3380     0000     2654     2655 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASRES   $ALLOC$     A01     3380     0000     0000     0001
         $DIRECT$    A04     3380     0001     0002     0002
         $CP-NUC$    A09     3380     0003     0012     0010
         $SYSCKP$    A06     3380     0013     0014     0002
         $SYSWRM$    A07     3380     0015     0016     0002
         $PAGE$      A03     3380     0017     0040     0024
         $SPOOL$     B01     3380     0041     0127     0087
         MAINT       392     3380     0128     0147     0020
         MAINT       492     3380     0148     0157     0010
         MAINT       491     3380     0158     0165     0008
         MAINT       423     3380     0166     0175     0010
         MAINT       193     3380     0176     0221     0046
         MAINT       791     3380     0222     0233     0012
         MAINT       190     3380     0234     0305     0072
         MAINT       391     3380     0306     0320     0015
         MAINT       291     3380     0321     0335     0015
         MAINT       194     3380     0336     0391     0056
         MAINT       293     3380     0392     0531     0140
         LGLOPR      191     3380     0532     0532     0001
         CMSBATCH    195     3380     0533     0534     0002
                                       535      536        2    GAP
         MAINT       36E     3380     0537     0538     0002
         $T-DISK$    B01     3380     0539     0558     0020
```

Figure 63 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3380-K

MAINT	691	3380	0559	0668	0110	
GCS	191	3380	0669	0672	0004	
MAINT	592	3380	0673	0682	0010	
			683	692	10	GAP
DISKACNT	191	3380	0693	0693	0001	
MAINT	191	3380	0694	0723	0030	
OP1	191	3380	0724	0724	0001	
OPERATOR	191	3380	0725	0729	0005	
OPERATNS	191	3380	0730	0746	0017	
MAINT	49C	3380	0747	0786	0040	
MAINT	201	3380	0787	0808	0022	
MAINT	292	3380	0809	0828	0020	
MAINT	395	3380	0829	0838	0010	
AUTOLOG1	191	3380	0839	0839	0001	
IBMPSR	191	3380	0840	0840	0001	
EREP	191	3380	0841	0842	0002	
			843	850	8	GAP
MAINT	692	3380	0851	0860	0010	
MAINT	593	3380	0861	0950	0090	
MAINT	594	3380	0951	1025	0075	
MAINT	295	3380	1026	1038	0013	
			1039	1092	54	GAP
MAINT	294	3380	1093	1212	0120	
MAINT	394	3380	1213	1342	0130	
MAINT	19C	3380	1343	1382	0040	
MAINT	596	3380	1383	1402	0020	
MAINT	59E	3380	1403	1412	0010	
			1413	1422	10	GAP
MAINT	495	3380	1423	1427	0005	
MAINT	892	3380	1428	1432	0005	
MAINT	490	3380	1433	1504	0072	
			1505	1529	25	GAP
MAINT	89E	3380	1530	1539	0010	
MAINT	896	3380	1540	1549	0010	
MAINT	895	3380	1550	1559	0010	
MAINT	595	3380	1560	1571	0012	
MAINT	5E5	3380	1572	1576	0005	
MAINT	591	3380	1577	1671	0095	
MAINT	501	3380	1672	1673	0002	
MAINT	19E	3380	1674	1717	0044	
			1718	1818	101	GAP
MAINT	393	3380	1819	1884	0066	
MAINT	49D	3380	1885	1924	0040	
MAINT	19D	3380	1925	1964	0040	
MAINT	123	3380	0000	2654	2655	**OVERLAP**
SYSDUMP1	123	3380	0000	2654	2655	**OVERLAP**

Figure 63 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3380-K

Sample Files for a 3390 Starter System

IBM provides the following sample files with the VM/XA SP 3390 Starter System:

- HCPSYS ASSEMBLE for Starter System
- System directories for 3390 and 3391
- System residence DASD allocation
- Minidisk maps of the system residence device.

Sample HCPSYS ASSEMBLE for a 3390

This is only a sample file. You need to tailor it to your installation before it can be used.

```
SYS TITLE 'HCPSYS90 - HCPSYS FOR VM/XA INSTALL BUILD'
*********************************************************************
* 5664-308 (C) COPYRIGHT IBM CORPORATION -      1989                *
* LICENSED MATERIALS - PROPERTY OF IBM                              *
* SEE COPYRIGHT INSTRUCTIONS, G120-2083                             *
*********************************************************************
*****************************************************************
*
* MODULE NAME - HCPSYS
*
* DESCRIPTIVE NAME - DEFINITION OF SYSTEM
*
*
* FUNCTION - TO DEFINE THE SYSTEM
*
*
*   REGISTER CONVENTIONS - SYMBOLIC REFERENCES TO REGISTERS ARE
*                          OF THE FORM "RX" WHERE X IS A NUMBER
*                          RANGING FROM 0 TO 15 (SEE HCPEQUAT COPY)
*
*   PATCH LABEL - NONE
*
* MODULE TYPE - DATA
*
*   PROCESSOR - ASSEMBLER H (VERSION 2)
*
*   ATTRIBUTES - RESIDENT, DATA-ONLY
*
* MACROS -
*
*          SYSSTORE - DEFINE SYSTEM REAL STORAGE
*          SYSRES   - DEFINE SYSTEM RESIDENCE VOLUME
*          SYSCPVOL - DEFINE SYSTEM VOLUMES
*          SYSOPR   - DEFINE SYSTEM OPERATOR
*          SYSTIME  - DEFINE THE TIME ZONE
*          SYSACNT  - DEFINE ACCOUNTING USERID
*          SYSDUMP  - DEFINE DUMP RECEIVER USERID
*          SYSEREP  - DEFINE EREP RECEIVER USERID
*          SYSID    - DEFINE SYSTEM ID'S BY PROCESSOR
*          SYSUVOL  - DEFINE USER VOLUMES TO BE MOUNTED
*          SYSFCN   - DEFINE PRIV CLASSES ASSIGNED TO CP FUNCTIONS
*          SYSEND   - END SYSTEM GENERATION
*
*
* GENERAL COMMENTS - THIS MODULE IS THE USER'S RESPONSIBILITY.
```

```
*
*************************************************************************
         SPACE 1
*-------------------------------------------------------------
* THE SYSTEM DEFINITION MACROS SYSSTORE, SYSRES, SYSCPVOL,
* SYSOPR, AND SYSTIME, ARE CODED HERE, IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSSTORE RMSIZE=32M,VRSIZE=8M,VRFREE=256
         SYSRES SYSVOL=XASRES,                                        *
               SYSRES=123,SYSTYPE=3390,                               *
               SYSNUC=003,                                            *
               SYSCKP=(013,2),                                        *
               SYSWRM=(015,2)
         SYSTIME ZONE=5,LOC=WEST,ID=EST
         SYSOPR SYSOPER=OPERATOR
         SYSCPVOL     XASRES,XASERV,XAP001,                           *
               DRUM01,DRUM02,DRUM03,DRUM04,DRUM05,DRUM06,DRUM07,      *
               PAG001,PAG002,PAG003,PAG004,PAG005,PAG006,PAG007,      *
               SPL001,SPL002,SPL003,SPL004,SPL005,SPL006,SPL007
         SPACE 1
*-------------------------------------------------------------
* ANY OF THE OPTIONAL MACROS THAT ARE DESIRED ARE CODED HERE,
* IN ANY ORDER
*-------------------------------------------------------------
         SPACE 1
         SYSID      DEFAULT=INSTTST,(3081A,3081,10000)
         SYSACNT USERID=DISKACNT
         SYSDUMP USERID=OPERATNS
         SYSEREP USERID=EREP
         SYSUVOL      USRP01,USRP02,USRP03,USRP04,USRP05,USRP06,USRP07,*
               USRP08,USRP09
         SYSFCN  OPER=A,CPRD=CE,CPWT=C,SERV=F,DFLT=G
         SPACE 1
*-------------------------------------------------------------
* THE SYSEND MACRO IS CODED HERE
*-------------------------------------------------------------
         SPACE 1
         SYSEND
```

There are two sets of sample directory, minidisk allocations, and minidisk maps for different models of 3390 DASD. The first set ("Sample Directory for a 3390," "System Residence DASD Allocation for a 3390" on page 834 , and "Minidisk Maps for a 3390 System Residence Device" on page 835) is for the 3390. The second set ("Sample Directory for a 3391" on page 837, "System Residence DASD Allocation for a 3391" on page 847, and "Minidisk Maps for a 3391 System Residence Device" on page 848,) is for the 3391.

Sample Directory for a 3390

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1989      *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM   *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083        *
*************************************************************
*************************************************************
*  3390 SYSTEM   VM/XA SYSTEM  PRODUCT  DIRECTORY           *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.   *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST  *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,   *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.   *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL  *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH    *
*  YOUR HARDWARE ADDRESSES.                                 *
*                                                           *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO  *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE  *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY  *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                 *
*                                                           *
*      USER ANYONE SOMEPASS 3M 8M FG                        *
*       AUTOLOG AUTOLOG1 OP1 MAINT                          *
*        ACCOUNT ??????                                     *
*        IPL 190                                            *
*        CONSOLE 01F 3215                                   *
*        SPOOL 00C 2540 READER A                            *
*        SPOOL 00D 2540 PUNCH A                             *
*        SPOOL 00E 1403 A                                   *
*===>    MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS  *
*                                                           *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                READ ONLY MODE.                            *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                WRITE MODE.                                *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN  *
*                MULTI-WRITE MODE.                          *
*                                                           *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.   *
*                                                           *
*************************************************************
*                                                           *
*                                                           *
*  Following is a sample user entry and a brief description  *
*  of each entry.  A more detailed description may be found  *
*  by referencing the VM/XA System Product Planning and     *
*  Administration manual.                                   *
*                                                           *
*    USER CMS1  abc123  3M  32M G                           *
```

```
*       - Userid is CMS1                                        *
*       - Logon password is abc123                              *
*       - At logon time, virtual storage will equal 3M          *
*       - Maximum storage that may be defined equals 32M        *
*       - Privilege class is G                                  *
*                                                               *
*     AUTOLOG AUTOLOG1 OP1 MAINT                                *
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,         *
*         OP1, and MAINT                                        *
*                                                               *
*     ACCOUNT ACT4 CMSTST                                       *
*       - Time used by CMS1 will be charged to account ACT4     *
*       - Printed output will be sent to distribution CMSTST    *
*                                                               *
*     IPL CMS                                                   *
*       - Will cause automatic IPL of named save system of      *
*         CMS                                                   *
*                                                               *
*     CONSOLE 009 3215                                          *
*       - Defines virtual machines console as a 3215 at a       *
*         virtual address of 009                                *
*                                                               *
*     SPOOL 00C 2540 READER A                                   *
*     SPOOL 00D 2540 PUNCH A                                    *
*     SPOOL 00E 1403 A                                          *
*       - Defines virtual unit record devices of reader,        *
*         punch, and printer with a spooling class of 'A'       *
*                                                               *
*     LINK MAINT 190 190 RR                                     *
*       - Allows read access to minidisk 190 of user 'MAINT'    *
*                                                               *
*     LINK MAINT 19E 19E RR                                     *
*       - Allows read access to minidisk 19E of user 'MAINT'    *
*                                                               *
*     LINK MAINT 19D 19D RR                                     *
*       - Allows read access to minidisk 19d of user 'MAINT'    *
*                                                               *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS              *
*       - Defines minidisk with a virtual address of 999        *
*       - Specifies that the minidisk resides on a real 3350    *
*       - Minidisk starts at cylinder 236                       *
*       - Mdisk is 22 cylinders in size (starting at cyl 236    *
*       - The real volume serial number is 'CPPACK'             *
*       - Mdisk is to be accessed in write mode if no other     *
*         user has write access. Alternate access read-only     *
*       - RPASS is the required password for another user to    *
*         link to this minidisk in read mode                    *
*       - WPASS is the required password for another user to    *
*         link to this minidisk in write mode                   *
*****************************************************************
*
*
*
DIRECTORY 123 3390 XASRES
*
*****************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS               *
*****************************************************************
*
USER $ALLOC$  NOLOG
```

```
 MDISK A01 3390 000 001 XASRES R
 MDISK B01 3390 000 001 XASERV R
 MDISK C01 3390 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
 MDISK A04 3390 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3390 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3390 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3390 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3390 017 023 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3390 040 082 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3390 406 019 XASRES R       ###
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                  PROP LOGICAL OPERATOR                            *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3390 599 001 XASRES WR      ###
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
```

```
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
 NAMESAVE GCS
 NAMESAVE VTAM
 NAMESAVE HELP
 NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3390 0000 1113 XASRES  MW
 MDISK 124 3390 0000 1113 XAP001  RR
 MDISK 125 3390 0000 1113 XASERV  RR

 MDISK 190 3390 0488 068 XASRES  MW ALL
 MDISK 191 3390 0444 028 XASRES  MW
 MDISK 193 3390 0277 043 XASRES  MW
 MDISK 194 3390 0210 053 XASRES  MW
 MDISK 5E5 3390 0594 005 XASRES  MW ALL
 MDISK 19D 3390 0320 038 XASRES  MW ALL
 MDISK 19E 3390 0364 042 XASRES  MW ALL
 MDISK 291 3390 0604 014 XASRES  MW
 MDISK 292 3390 0618 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 391 3390 0263 014 XASRES  MW
 MDISK 392 3390 0708 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 423 3390 0200 010 XASRES  MW
 MDISK 490 3390 0132 068 XASRES  MW
 MDISK 492 3390 0727 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 49D 3390 0556 038 XASRES  MW READ WRITE MULTIPLE
 MDISK 593 3390 0747 085 XASRES  MW READ WRITE MULTIPLE
 MDISK 594 3390 0637 071 XASRES  MW READ WRITE MULTIPLE
 MDISK 692 3390 0737 010 XASRES  MW READ WRITE MULTIPLE


 MDISK 19C 3390 0330 038 XASERV  WR
 MDISK 201 3390 0001 021 XASERV  MW RMAINT   WMAINT   MMAINT
 MDISK 293 3390 0406 132 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3390 0538 113 XASERV  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 295 3390 0842 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 393 3390 0868 063 XASERV  WR
 MDISK 394 3390 0077 123 XASERV  WR
 MDISK 395 3390 0832 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 491 3390 0200 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 495 3390 0208 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 49C 3390 0368 038 XASERV  WR
 MDISK 501 3390 0213 002 XASERV  MW READ WRITE MULTIPLE
 MDISK 591 3390 0674 090 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3390 0215 019 XASERV  MW READ WRITE MULTIPLE
 MDISK 595 3390 0234 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 596 3390 0246 019 XASERV  WR
 MDISK 59E 3390 0265 012 XASERV  WR
 MDISK 691 3390 0764 104 XASERV  MW READ WRITE MULTIPLE
 MDISK 791 3390 0277 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 892 3390 0289 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 895 3390 0306 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 896 3390 0318 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 89E 3390 0294 012 XASERV  MW READ WRITE MULTIPLE
```

```
*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3390 0022 002 XASERV RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3390 602 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3390 358 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT NOLOG    3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
  IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 122 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
```

```
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 3 OPERATOR
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 129 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3390 000 1113 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
```

```
  MDISK 191 3390 127 002 XASRES WR READ     WRITE    MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 472 016 XASRES MR RDVF     WDVF     MDVF
*
USER IPCS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 8 CE-ROOM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK OPERATNS 193 192 RR
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3390 600 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 601 001 XASRES MR READ WRITE MULTIPLE
*
USER IBMPSR NOLOG  3M 8M BG
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
```

```
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 130 001 XASRES WR
*
USER IVPM2 NOLOG  3M   4M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT5 IVPM2
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3390 000 1113 XASRES RR
 MDISK 124 3390 000 1113 XAP001 RR
 MDISK 125 3390 000 1113 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
```

```
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3390

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3390 0000 1113 XASRES  MW
 MDISK 124 3390 0000 1113 XAP001  RR
 MDISK 125 3390 0000 1113 XASERV  RR

 MDISK 190 3390 0488 068 XASRES  MW ALL
 MDISK 191 3390 0444 028 XASRES  MW
 MDISK 193 3390 0277 043 XASRES  MW
 MDISK 194 3390 0210 053 XASRES  MW
 MDISK 5E5 3390 0594 005 XASRES  MW ALL
 MDISK 19D 3390 0320 038 XASRES  MW ALL
 MDISK 19E 3390 0364 042 XASRES  MW ALL
 MDISK 291 3390 0604 014 XASRES  MW
 MDISK 292 3390 0618 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 391 3390 0263 014 XASRES  MW
 MDISK 392 3390 0708 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 423 3390 0200 010 XASRES  MW
 MDISK 490 3390 0132 068 XASRES  MW
 MDISK 492 3390 0727 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 49D 3390 0556 038 XASRES  MW READ WRITE MULTIPLE
 MDISK 593 3390 0747 085 XASRES  MW READ WRITE MULTIPLE
 MDISK 594 3390 0637 071 XASRES  MW READ WRITE MULTIPLE
 MDISK 692 3390 0737 010 XASRES  MW READ WRITE MULTIPLE


 MDISK 19C 3390 0330 038 XASERV  WR
 MDISK 201 3390 0001 021 XASERV  MW RMAINT   WMAINT   MMAINT
 MDISK 293 3390 0406 132 XASERV  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3390 0538 113 XASERV  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 295 3390 0842 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 393 3390 0868 063 XASERV  WR
 MDISK 394 3390 0077 123 XASERV  WR
 MDISK 395 3390 0832 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 491 3390 0200 008 XASERV  MW READ WRITE MULTIPLE
 MDISK 495 3390 0208 005 XASERV  MW READ WRITE MULTIPLE
 MDISK 49C 3390 0368 038 XASERV  WR
 MDISK 501 3390 0213 002 XASERV  MW READ WRITE MULTIPLE
 MDISK 591 3390 0674 090 XASERV  MW READ WRITE MULTIPLE
 MDISK 592 3390 0215 019 XASERV  MW READ WRITE MULTIPLE
 MDISK 595 3390 0234 012 XASERV  MW READ WRITE MULTIPLE
 MDISK 596 3390 0246 019 XASERV  WR
 MDISK 59E 3390 0265 012 XASERV  WR
```

```
MDISK 691 3390 0764 104 XASERV  MW READ WRITE MULTIPLE
MDISK 791 3390 0277 012 XASERV  MW READ WRITE MULTIPLE
MDISK 892 3390 0289 005 XASERV  MW READ WRITE MULTIPLE
MDISK 895 3390 0306 012 XASERV  MW READ WRITE MULTIPLE
MDISK 896 3390 0318 012 XASERV  MW READ WRITE MULTIPLE
MDISK 89E 3390 0294 012 XASERV  MW READ WRITE MULTIPLE

*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3390 0022 002 XASERV RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
 LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3390

Figure 64 shows the allocation for the system residence device for the 3390 VM/XA SP starter system. The allocation and minidisk maps for the 3391 ("System Residence DASD Allocation for a 3391" on page 847 and "Minidisk Maps for a 3391 System Residence Device" on page 848) follow the corresponding sample directories ("Sample Directory for a 3391" on page 837 and "Sample XAMAINT Directory Entry for a 3391" on page 845, respectively).

```
TYPE      CYL  CYL
PERM      0000-0000
DRCT      0001-0002
PERM      0003-0016
PAGE      0017-0039
SPOL      0040-0121
PERM      0122 0405
TDSK      0406-0424
PERM      0425-1113
```

Figure 64. System Residence DASD Allocation for a 3390

Minidisk Maps for a 3390 System Residence Device

Figure 65 shows the minidisk maps of the system residence device for the 3390. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3390     0000     1112     1113

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3390     0000     0000     0001
         MAINT       124     3390     0000     1112     1113 **OVERLAP**
         SYSDUMP1    124     3390     0000     1112     1113 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3390     0000     0000     0001
         MAINT       201     3390     0001     0021     0021
         MAINT       36E     3390     0022     0023     0002
                                        24       76       53    GAP
         MAINT       394     3390     0077     0199     0123
         MAINT       491     3390     0200     0207     0008
         MAINT       495     3390     0208     0212     0005
         MAINT       501     3390     0213     0214     0002
         MAINT       592     3390     0215     0233     0019
         MAINT       595     3390     0234     0245     0012
         MAINT       596     3390     0246     0264     0019
         MAINT       59E     3390     0265     0276     0012
         MAINT       791     3390     0277     0288     0012
         MAINT       892     3390     0289     0293     0005
         MAINT       89E     3390     0294     0305     0012
         MAINT       895     3390     0306     0317     0012
         MAINT       896     3390     0318     0329     0012
         MAINT       19C     3390     0330     0367     0038
         MAINT       49C     3390     0368     0405     0038
         MAINT       293     3390     0406     0537     0132
         MAINT       294     3390     0538     0650     0113
                                       651      673       23    GAP
         MAINT       591     3390     0674     0763     0090
         MAINT       691     3390     0764     0867     0104
         MAINT       393     3390     0868     0930     0063
         MAINT       125     3390     0000     1112     1113 **OVERLAP**
         SYSDUMP1    125     3390     0000     1112     1113 **OVERLAP**
```

Figure 65 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3390

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START     END     SIZE
XASRES   $ALLOC$     A01     3390     0000     0000     0001
         $DIRECT$    A04     3390     0001     0002     0002
         $CP-NUC$    A09     3390     0003     0012     0010
         $SYSCKP$    A06     3390     0013     0014     0002
         $SYSWRM$    A07     3390     0015     0016     0002
         $PAGE$      A03     3390     0017     0039     0023
         $SPOOL$     B01     3390     0040     0121     0082
         OPERATOR    191     3390     0122     0126     0005
         EREP        191     3390     0127     0128     0002
         OP1         191     3390     0129     0129     0001
         IBMPSR      191     3390     0130     0130     0001
                                       131      131        1    GAP
         MAINT       490     3390     0132     0199     0068
         MAINT       423     3390     0200     0209     0010
         MAINT       194     3390     0210     0262     0053
         MAINT       391     3390     0263     0276     0014
         MAINT       193     3390     0277     0319     0043
         MAINT       19D     3390     0320     0357     0038
         GCS         191     3390     0358     0361     0004
                                       362      363        2    GAP
         MAINT       19E     3390     0364     0405     0042
         $T-DISK$    B01     3390     0406     0424     0019
                                       425      443       19    GAP
         MAINT       191     3390     0444     0471     0028
         OPERATNS    191     3390     0472     0487     0016
         MAINT       190     3390     0488     0555     0068
         MAINT       49D     3390     0556     0593     0038
         MAINT       5E5     3390     0594     0598     0005
         LGLOPR      191     3390     0599     0599     0001
         AUTOLOG1    191     3390     0600     0600     0001
         DISKACNT    191     3390     0601     0601     0001
         CMSBATCH    195     3390     0602     0603     0002
         MAINT       291     3390     0604     0617     0014
         MAINT       292     3390     0618     0636     0019
         MAINT       594     3390     0637     0707     0071
         MAINT       392     3390     0708     0726     0019
         MAINT       492     3390     0727     0736     0010
         MAINT       692     3390     0737     0746     0010
         MAINT       593     3390     0747     0831     0085
         MAINT       395     3390     0832     0841     0010
         MAINT       295     3390     0842     0854     0013
         MAINT       123     3390     0000     1112     1113 **OVERLAP**
         SYSDUMP1    123     3390     0000     1112     1113 **OVERLAP**
```

Figure 65 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3390

Sample Directory for a 3391

This is only a sample file. You need to tailor it to your installation before it can be used.

```
*****************************************************************
* COPYRIGHT - 5664-308 - (c) COPYRIGHT IBM CORP.- 1989          *
*             LICENSED MATERIAL - PROGRAM PROPERTY OF IBM       *
*             SEE COPYRIGHT INSTRUCTIONS,  G120-2083            *
*****************************************************************
*****************************************************************
*  3390 SYSTEM   VM/XA SYSTEM  PRODUCT  DIRECTORY               *
*  THE ADDRESSES 123, 124, AND 125 ARE VIRTUAL ADDRESSES.       *
*  THE ADDRESS 123 IS CRITICAL SINCE IT IS USED IN HCPSYS AND   *
*  THE DIRECTORY. IF YOU WANT TO CHANGE IT, REMEMBER IT MUST    *
*  BE CHANGED IN HCPSYS, THE 'DIRECTORY' STATEMENT BELOW,       *
*  AND IN THE 'MDISK' STATEMENTS FOR THE USERID  'MAINT'.       *
*  NOTE: REMEMBER THESE ARE ONLY VIRTUAL ADDRESSES NOT REAL     *
*  ADDRESSES, SO THERE IS NO NEED TO CHANGE THEM TO MATCH       *
*  YOUR HARDWARE ADDRESSES.                                     *
*                                                               *
*  THIS DIRECTORY MUST BE MODIFIED IF MDISK PASSWORDS ARE TO    *
*  BE SPECIFIED.  WITHOUT MDISK PASSWORDS, MDISKS CANNOT BE     *
*  SHARED USING THE 'CP LINK' COMMAND.  A SAMPLE USER ENTRY     *
*  FOLLOWS: (NOTE ONLY THE MDISK STATEMENT)                     *
*                                                               *
*      USER ANYONE SOMEPASS 3M 8M FG                            *
*       AUTOLOG AUTOLOG1 OP1 MAINT                              *
*        ACCOUNT ??????                                         *
*        IPL 190                                                *
*        CONSOLE 01F 3215                                       *
*        SPOOL 00C 2540 READER A                                *
*        SPOOL 00D 2540 PUNCH A                                 *
*        SPOOL 00E 1403 A                                       *
*====>   MDISK 999 33XX XXX YYY ZZZZZZ MR RPASS WPASS MPASS     *
*                                                               *
*    WHERE:  RPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                READ ONLY MODE.                                *
*            WPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                WRITE MODE.                                    *
*            MPASS WILL BE THE REQUIRED PASSWORD TO LINK IN     *
*                MULTI-WRITE MODE.                              *
*                                                               *
*   NOTE:  EACH PASSWORD MAY BE 1-8 CHARACTERS IN LENGTH.       *
*                                                               *
*****************************************************************
*                                                               *
*                                                               *
*  Following is a sample user entry and a brief description     *
*  of each entry.  A more detailed description may be found     *
*  by referencing the VM/XA System Product Planning and         *
*  Administration manual.                                       *
*                                                               *
*    USER CMS1  abc123  3M  32M G                               *
*       - Userid is CMS1                                        *
*       - Logon password is abc123                              *
*       - At logon time, virtual storage will equal 3M          *
*       - Maximum storage that may be defined equals 32M        *
*       - Privilege class is G                                  *
*                                                               *
*     AUTOLOG AUTOLOG1 OP1 MAINT                                *
```

```
*       - Allows user CMS1 to be autolog'd by AUTOLOG1,      *
*         OP1, and MAINT                                     *
*                                                            *
*     ACCOUNT ACT4 CMSTST                                    *
*       - Time used by CMS1 will be charged to account ACT4  *
*       - Printed output will be sent to distribution CMSTST *
*                                                            *
*     IPL CMS                                                *
*       - Will cause automatic IPL of named save system of   *
*         CMS                                                *
*                                                            *
*     CONSOLE 009 3215                                       *
*       - Defines virtual machines console as a 3215 at a    *
*         virtual address of 009                             *
*                                                            *
*     SPOOL 00C 2540 READER A                                *
*     SPOOL 00D 2540 PUNCH A                                 *
*     SPOOL 00E 1403 A                                       *
*       - Defines virtual unit record devices of reader,     *
*         punch, and printer with a spooling class of 'A'    *
*                                                            *
*     LINK MAINT 190 190 RR                                  *
*       - Allows read access to minidisk 190 of user 'MAINT' *
*                                                            *
*     LINK MAINT 19E 19E RR                                  *
*       - Allows read access to minidisk 19E of user 'MAINT' *
*                                                            *
*     LINK MAINT 19D 19D RR                                  *
*       - Allows read access to minidisk 19d of user 'MAINT' *
*                                                            *
*     MDISK 999 3350 236 022 CPPACK WR RPASS WPASS           *
*       - Defines minidisk with a virtual address of 999     *
*       - Specifies that the minidisk resides on a real 3350 *
*       - Minidisk starts at cylinder 236                    *
*       - Mdisk is 22 cylinders in size (starting at cyl 236 *
*       - The real volume serial number is 'CPPACK'          *
*       - Mdisk is to be accessed in write mode if no other  *
*         user has write access. Alternate access read-only  *
*       - RPASS is the required password for another user to *
*         link to this minidisk in read mode                 *
*       - WPASS is the required password for another user to *
*         link to this minidisk in write mode                *
*************************************************************
*
*
*
DIRECTORY 123 3390 XASRES
*
*************************************************************
*         SYSTEM RESERVED AREAS NOT FOR MINIDISKS           *
*************************************************************
*
USER $ALLOC$  NOLOG
 MDISK A01 3390 000 001 XASRES R
 MDISK B01 3390 000 001 XASERV R
 MDISK C01 3390 000 001 XAP001 R
*
USER $DIRECT$ NOLOG
```

```
 MDISK A04 3390 001 002 XASRES  R
*
USER $CP-NUC$ NOLOG
 MDISK A09 3390 003 010 XASRES  R
*
USER $SYSCKP$ NOLOG
 MDISK A06 3390 013 002 XASRES  R
*
USER $SYSWRM$ NOLOG
 MDISK A07 3390 015 002 XASRES  R
*
USER $PAGE$   NOLOG
 MDISK A03 3390 017 023 XASRES R
*
USER $SPOOL$  NOLOG
 MDISK B01 3390 040 082 XASRES R
*
USER $T-DISK$ NOLOG
 MDISK B01 3390 406 019 XASRES R
*
*****************************************************************
*             CMS USERIDS                                       *
*****************************************************************
*
USER CMS1  NOLOG  3M   32M G
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT ACT4 CMSTST
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
*********************************************************************
*                  PROP LOGICAL OPERATOR                            *
*********************************************************************
*
USER LGLOPR  NOLOG 512K 16M ABCDEG
 ACCOUNT ACT1 PROP
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH A
 SPOOL 00E 1403 A
 MDISK 191 3390 599 001 XASRES WR
 LINK MAINT 194 194 RR
 LINK MAINT 190 190 RR
 LINK OPERATOR 191 291 RR
*
*****************************************************************
*             SYSTEM RELATED USERIDS                            *
*****************************************************************
*
USER MAINT NOLOG 16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
 IPL 190
```

```
NAMESAVE GCS
NAMESAVE VTAM
NAMESAVE HELP
NAMESAVE INSTHELP
CONSOLE 009 3215 T
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
MDISK 123 3390 0000 2226 XASRES  MW
MDISK 124 3390 0000 2226 XAP001  RR
MDISK 125 3390 0000 2226 XASERV  RR

MDISK 190 3390 0488 068 XASRES  MW ALL
MDISK 191 3390 0444 028 XASRES  MW
MDISK 193 3390 0277 043 XASRES  MW
MDISK 194 3390 0210 053 XASRES  MW
MDISK 5E5 3390 0594 005 XASRES  MW ALL
MDISK 19D 3390 0320 038 XASRES  MW ALL
MDISK 19E 3390 0364 042 XASRES  MW ALL
MDISK 291 3390 0604 014 XASRES  MW
MDISK 292 3390 0618 019 XASRES  MW READ WRITE MULTIPLE
MDISK 391 3390 0263 014 XASRES  MW
MDISK 392 3390 0708 019 XASRES  MW READ WRITE MULTIPLE
MDISK 423 3390 0200 010 XASRES  MW
MDISK 490 3390 0132 068 XASRES  MW
MDISK 492 3390 0727 010 XASRES  MW READ WRITE MULTIPLE
MDISK 49D 3390 0556 038 XASRES  MW READ WRITE MULTIPLE
MDISK 593 3390 0747 085 XASRES  MW READ WRITE MULTIPLE
MDISK 594 3390 0637 071 XASRES  MW READ WRITE MULTIPLE
MDISK 692 3390 0737 010 XASRES  MW READ WRITE MULTIPLE


MDISK 19C 3390 1161 038 XASRES  WR
MDISK 201 3390 0832 021 XASRES  MW RMAINT   WMAINT   MMAINT
MDISK 293 3390 1237 132 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
MDISK 294 3390 1369 113 XASRES  MW RCPAUX   WCPAUX   MCPAUX
MDISK 295 3390 1482 013 XASRES  MW READ WRITE MULTIPLE
MDISK 393 3390 1699 063 XASRES  WR
MDISK 394 3390 0908 123 XASRES  WR
MDISK 395 3390 1495 010 XASRES  MW READ WRITE MULTIPLE
MDISK 491 3390 1031 008 XASRES  MW READ WRITE MULTIPLE
MDISK 495 3390 1039 005 XASRES  MW READ WRITE MULTIPLE
MDISK 49C 3390 1199 038 XASRES  WR
MDISK 501 3390 1044 002 XASRES  MW READ WRITE MULTIPLE
MDISK 591 3390 1505 090 XASRES  MW READ WRITE MULTIPLE
MDISK 592 3390 1046 019 XASRES  MW READ WRITE MULTIPLE
MDISK 595 3390 1065 012 XASRES  MW READ WRITE MULTIPLE
MDISK 596 3390 1077 019 XASRES  WR
MDISK 59E 3390 1096 012 XASRES  WR
MDISK 691 3390 1595 104 XASRES  MW READ WRITE MULTIPLE
MDISK 791 3390 1108 012 XASRES  MW READ WRITE MULTIPLE
MDISK 892 3390 1120 005 XASRES  MW READ WRITE MULTIPLE
MDISK 895 3390 1137 012 XASRES  MW READ WRITE MULTIPLE
MDISK 896 3390 1149 012 XASRES  MW READ WRITE MULTIPLE
MDISK 89E 3390 1125 012 XASRES  MW READ WRITE MULTIPLE

*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3390 0853 002 XASRES RR RPVM     WPVM     MPVM
```

```
 LINK RSCS  191 499  MW
*
USER CMSBATCH NOLOG 1M 2M G
 ACCOUNT 3 SYSTEM
 OPTION ACCT
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 195 3390 602 002 XASRES MR RBATCH   WBATCH   MBATCH
*
USER GCS NOLOG 16M 16M ABCDEFG
  AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT GCS RECVM
  IPL GCS
  NAMESAVE GCS
  CONSOLE 009 3215
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 595 595
  LINK MAINT 59E 59E
  MDISK 191 3390 358 004 XASRES MR RGCS WGCS MGCS
*
USER SYSMAINT NOLOG    3M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 SYSPROG
  IPL CMS
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 123 123  MW
 LINK MAINT 191 192  RR
 LINK MAINT 190 190  RR
 LINK MAINT 19E 19E  RR
 LINK MAINT 19D 19D  RR
*
USER OPERATOR NOLOG   16M 32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 2 OPERATOR
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 122 005 XASRES MR
 LINK OP1   191 192 RR
*
*OP1 IS AN ALTERNATE OPERATOR USERID
*
USER OP1 NOLOG 3M  32M ABCDEFG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 3 OPERATOR
```

```
 IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 129 001 XASRES MR
 LINK OPERATOR 191 192 RR
*
* This userid is for the IBM CE's use in running
* OLTSEP.
* OLTSEP is automatically IPLed in the virtual machine.
* A minimum machine size of one megabyte is required to
* run OLTSEP.
* The privilege class of F allows the CE to specify
* intensive recording mode.
* The console address of 01F is required by OLTSEP.
* The unit record addresses are those required by
* OLTSEP.
* The 5FF minidisk is the CE's OLTSEP pack.
*
USER OLTSEP NOLOG 3M 8M FG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT OLTSEP IBMCE
  IPL 5FF
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH A
  SPOOL 00E 1403 A
  MDISK 5FF 3390 000 2226 CEPACK MR
*
*       **  EREP **
*
* This userid is for the IBM CE's use in running
* CPEREP.
* CMS is automatically IPLed in the virtual machine.
* The 190 minidisk is the CMS system disk.
* The 191 minidisk may be used to save often-used EREP
* control statements and procedures (EXECs).
* THE 190 MINIDISK HAS THE CMS TXTLIBS NECESSARY TO RUN
* EREP.
*
USER EREP NOLOG 3M 8M BFG
 AUTOLOG AUTOLOG1 OP1 MAINT
  ACCOUNT EREP IBMCE
   IPL CMS
  CONSOLE 01F 3215
  SPOOL 00C 2540 READER A
  SPOOL 00D 2540 PUNCH B
  SPOOL 00E 1403 A
  LINK MAINT 190 190 RR
  LINK MAINT 201 201 RR
  MDISK 191 3390 127 002 XASRES WR READ     WRITE   MULTIPLE
*
USER OPERATNS NOLOG 3M 8M BCEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 1 OPERATNS
 IPL CMS
```

```
 CONSOLE 009 3215
 SPOOL 00C 2540 READER D
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 472 016 XASRES MR RDVF     WDVF     MDVF
*
*
USER AUTOLOG1  NOLOG   3M 8M ABCDEG
 AUTOLOG OP1 MAINT
 ACCOUNT 9 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 MDISK 191 3390 600 001 XASRES MR RAUTOLOG WAUTOLOG MAUTOLOG
*
USER DISKACNT NOLOG 3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 10 ACCNTNG
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 191 3390 601 001 XASRES MR READ WRITE MULTIPLE
*
USER VMUTIL NOLOG  3M 8M BDEG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 11 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
*
USER SYSDUMP1 NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 12 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 MDISK 123 3390 000 2226 XASRES RR
 MDISK 124 3390 000 2226 XAP001 RR
```

```
 MDISK 125 3390 000 2226 XASERV RR
*
*
USER RSCS     NOLOG   3M 8M  BG
AUTOLOG AUTOLOG1 OP1 MAINT
ACCOUNT 13 SYSTEM
OPTION ACCT SVMSTAT LANG UCENG
IUCV ANY
NOPDATA
* INCLUDE IF GCS NAMED SAVED SYSTEM IS RESTRICTED
*NAMESAVE GCS
IPL GCS PARM AUTOLOG
CONSOLE 01F 3215 T OPERATOR
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 E
LINK MAINT 595 595 RR
LINK MAINT 59F 191 RR
LINK MAINT 190 190 RR
LINK MAINT 193 193 RR
LINK MAINT 19E 19E RR
* Assigns real devices or communication lines at the specified
* addresses to this virtual machine.
*DEDICATE 740 740
*
USER PVM      NOLOG   3M 8M BG
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT 14 SYSTEM
  IPL CMS
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19E 19E RR
 LINK MAINT 19D 19D RR
 LINK MAINT 36E 191 MR
*
*
*****************************************************************
*             OTHER OPERATING SYSTEM USERIDS                    *
*****************************************************************
*
```

Sample XAMAINT Directory Entry for a 3391

If you are using the procedure in Chapter 4, "Installing VM/XA System Product Release 2.1 Using an Existing VM/SP or VM/SP HPO System" on page 215 to install your system, add the following entry to your directory:

```
*
USER XAMAINT NOLOG 16M 16M ABCDEFG
  OPTION ECMODE DIAG98
 ACCOUNT 1 SYSPROG
 IPL 190
*NAMESAVE GCS
*NAMESAVE VTAM
*NAMESAVE HELP
*NAMESAVE INSTHELP
 CONSOLE 009 3215 T
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 MDISK 123 3390 0000 2226 XASRES  MW
 MDISK 124 3390 0000 2226 XAP001  RR
 MDISK 125 3390 0000 2226 XASERV  RR

 MDISK 190 3390 0488 068 XASRES  MW ALL
 MDISK 191 3390 0444 028 XASRES  MW
 MDISK 193 3390 0277 043 XASRES  MW
 MDISK 194 3390 0210 053 XASRES  MW
 MDISK 5E5 3390 0594 005 XASRES  MW ALL
 MDISK 19D 3390 0320 038 XASRES  MW ALL
 MDISK 19E 3390 0364 042 XASRES  MW ALL
 MDISK 291 3390 0604 014 XASRES  MW
 MDISK 292 3390 0618 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 391 3390 0263 014 XASRES  MW
 MDISK 392 3390 0708 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 423 3390 0200 010 XASRES  MW
 MDISK 490 3390 0132 068 XASRES  MW
 MDISK 492 3390 0727 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 49D 3390 0556 038 XASRES  MW READ WRITE MULTIPLE
 MDISK 593 3390 0747 085 XASRES  MW READ WRITE MULTIPLE
 MDISK 594 3390 0637 071 XASRES  MW READ WRITE MULTIPLE
 MDISK 692 3390 0737 010 XASRES  MW READ WRITE MULTIPLE


 MDISK 19C 3390 1161 038 XASRES  WR
 MDISK 201 3390 0832 021 XASRES  MW RMAINT   WMAINT   MMAINT
 MDISK 293 3390 1237 132 XASRES  MW RCMSAUX  WCMSAUX  MCMSAUX
 MDISK 294 3390 1369 113 XASRES  MW RCPAUX   WCPAUX   MCPAUX
 MDISK 295 3390 1482 013 XASRES  MW READ WRITE MULTIPLE
 MDISK 393 3390 1699 063 XASRES  WR
 MDISK 394 3390 0908 123 XASRES  WR
 MDISK 395 3390 1495 010 XASRES  MW READ WRITE MULTIPLE
 MDISK 491 3390 1031 008 XASRES  MW READ WRITE MULTIPLE
 MDISK 495 3390 1039 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 49C 3390 1199 038 XASRES  WR
 MDISK 501 3390 1044 002 XASRES  MW READ WRITE MULTIPLE
 MDISK 591 3390 1505 090 XASRES  MW READ WRITE MULTIPLE
 MDISK 592 3390 1046 019 XASRES  MW READ WRITE MULTIPLE
 MDISK 595 3390 1065 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 596 3390 1077 019 XASRES  WR
 MDISK 59E 3390 1096 012 XASRES  WR
```

```
 MDISK 691 3390 1595 104 XASRES  MW READ WRITE MULTIPLE
 MDISK 791 3390 1108 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 892 3390 1120 005 XASRES  MW READ WRITE MULTIPLE
 MDISK 895 3390 1137 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 896 3390 1149 012 XASRES  MW READ WRITE MULTIPLE
 MDISK 89E 3390 1125 012 XASRES  MW READ WRITE MULTIPLE

*
* MDISK STATEMENTS FOR INSTALL & SERVICE OF RSCS & PASS-THROUGH
 MDISK 36E 3390 0853 002 XASRES RR RPVM     WPVM     MPVM
 LINK RSCS  191 499  MW
 LINK MAINT 190 390  RR
```

System Residence DASD Allocation for a 3391

Figure 66 shows the allocation for the system residence device for the 3391 VM/XA SP Starter System.

```
TYPE      CYL  CYL
PERM      0000-0000
DRCT      0001-0002
PERM      0003-0016
PAGE      0017-0039
SPOL      0040-0121
PERM      0122 0405
TDSK      0406-0424
PERM      0425-1113
```

Figure 66. System Residence DASD Allocation for a 3391

Minidisk Maps for a 3391 System Residence Device

Figure 67 shows the minidisk maps of the system residence device for the 3391. To map your own system residence device for comparison, issue DISKMAP *fn* DIRECT, where *fn* is the name of your directory.

```
------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
CEPACK   OLTSEP      5FF     3390     0000     2225     2226

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XAP001   $ALLOC$     C01     3390     0000     0000     0001
         MAINT       124     3390     0000     2225     2226 **OVERLAP**
         SYSDUMP1    124     3390     0000     2225     2226 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASERV   $ALLOC$     B01     3390     0000     0000     0001
         MAINT       125     3390     0000     2225     2226 **OVERLAP**
         SYSDUMP1    125     3390     0000     2225     2226 **OVERLAP**

------------------------------------------------------------------------

VOLUME   USERID      CUU   DEVTYPE   START      END     SIZE
XASRES   $ALLOC$     A01     3390     0000     0000     0001
         $DIRECT$    A04     3390     0001     0002     0002
         $CP-NUC$    A09     3390     0003     0012     0010
         $SYSCKP$    A06     3390     0013     0014     0002
         $SYSWRM$    A07     3390     0015     0016     0002
         $PAGE$      A03     3390     0017     0039     0023
         $SPOOL$     B01     3390     0040     0121     0082
         OPERATOR    191     3390     0122     0126     0005
         EREP        191     3390     0127     0128     0002
         OP1         191     3390     0129     0129     0001
                                       130      131        2    GAP
         MAINT       490     3390     0132     0199     0068
         MAINT       423     3390     0200     0209     0010
         MAINT       194     3390     0210     0262     0053
         MAINT       391     3390     0263     0276     0014
         MAINT       193     3390     0277     0319     0043
         MAINT       19D     3390     0320     0357     0038
         GCS         191     3390     0358     0361     0004
                                       362      363        2    GAP
         MAINT       19E     3390     0364     0405     0042
         $T-DISK$    B01     3390     0406     0424     0019
                                       425      443       19    GAP
```

Figure 67 (Part 1 of 2). Minidisk Maps of the System Residence Device for a 3391

```
MAINT     191   3390   0444   0471   0028
OPERATNS  191   3390   0472   0487   0016
MAINT     190   3390   0488   0555   0068
MAINT     49D   3390   0556   0593   0038
MAINT     5E5   3390   0594   0598   0005
LGLOPR    191   3390   0599   0599   0001
AUTOLOG1  191   3390   0600   0600   0001
DISKACNT  191   3390   0601   0601   0001
CMSBATCH  195   3390   0602   0603   0002
MAINT     291   3390   0604   0617   0014
MAINT     292   3390   0618   0636   0019
MAINT     594   3390   0637   0707   0071
MAINT     392   3390   0708   0726   0019
MAINT     492   3390   0727   0736   0010
MAINT     692   3390   0737   0746   0010
MAINT     593   3390   0747   0831   0085
MAINT     201   3390   0832   0852   0021
MAINT     36E   3390   0853   0854   0002
                       855    907     53   GAP
MAINT     394   3390   0908   1030   0123
MAINT     491   3390   1031   1038   0008
MAINT     495   3390   1039   1043   0005
MAINT     501   3390   1044   1045   0002
MAINT     592   3390   1046   1064   0019
MAINT     595   3390   1065   1076   0012
MAINT     596   3390   1077   1095   0019
MAINT     59E   3390   1096   1107   0012
MAINT     791   3390   1108   1119   0012
MAINT     892   3390   1120   1124   0005
MAINT     89E   3390   1125   1136   0012
MAINT     895   3390   1137   1148   0012
MAINT     896   3390   1149   1160   0012
MAINT     19C   3390   1161   1198   0038
MAINT     49C   3390   1199   1236   0038
MAINT     293   3390   1237   1368   0132
MAINT     294   3390   1369   1481   0113
MAINT     295   3390   1482   1494   0013
MAINT     395   3390   1495   1504   0010
MAINT     591   3390   1505   1594   0090
MAINT     691   3390   1595   1698   0104
MAINT     393   3390   1699   1761   0063
MAINT     123   3390   0000   2225   2226 **OVERLAP**
SYSDUMP1  123   3390   0000   2225   2226 **OVERLAP**
```

Figure 67 (Part 2 of 2). Minidisk Maps of the System Residence Device for a 3391

Sample SPLOAD PROFILE

Information about the product tape layout, fields for userid, minidisk address, format, volume number and tape number are contained in this profile. The following is an example of the SPLOAD profile for the 3420 (6250 BPI) tape.

```
!-------------------------------------------------------------!
!             6250 BPI VM/XA Merged Product Tape              !
!-------------------------------------------------------------!
!
! Volume 1
!
Memo      Files      MAINT     191   XASP20V1   1        2
Install   Tools      MAINT     191   XASP20V1   1        3
Sysgen    Tools      MAINT     193   XASP20V1   1        4
System    Samples    MAINT     191   XASP20V1   1        5
CPLOC     Samples    MAINT     295   XASP20V1   1        6
CMSLOC    Samples    MAINT     395   XASP20V1   1        7
CP        Object     MAINT     194   XASP20V1   1        8
CP        Putapply   MAINT     192   XASP20V1   1        9
CP        Putdelta   MAINT     294   XASP20V1   1       10
CP        Corapply   MAINT     291   XASP20V1   1       11
CP        Cordelta   MAINT     291   XASP20V1   1       12
DUMPVIEW  Object     MAINT     193   XASP20V1   1       13
DUMPVIEW  Putapply   MAINT     392   XASP20V1   1       14
DUMPVIEW  Putdelta   MAINT     293   XASP20V1   1       15
DUMPVIEW  Corapply   MAINT     391   XASP20V1   1       16
DUMPVIEW  Cordelta   MAINT     391   XASP20V1   1       17
!
! Volume 2
!
CMS       Base       MAINT     193   XASP20V2   2        2
CMS       Putapply   MAINT     392   XASP20V2   2        3
CMS       Putdelta   MAINT     293   XASP20V2   2        4
CMS       Corapply   MAINT     391   XASP20V2   2        5
CMS       Cordelta   MAINT     391   XASP20V2   2        6
IOCP      Files      MAINT     193   XASP20V2   2        7
CMS       System     MAINT     190   XASP20V2   2        8
GCS       Object     MAINT     595   XASP20V2   2        9
GCS       Interface  MAINT     193   XASP20V2   2       10
GCS       Putapply   MAINT     592   XASP20V2   2       11
GCS       Putdelta   MAINT     596   XASP20V2   2       12
GCS       Corapply   MAINT     491   XASP20V2   2       13
GCS       Cordelta   MAINT     491   XASP20V2   2       14
AMENGHLP  Files      MAINT     19D   XASP20V2   2       15
UCENGHLP  Files      MAINT     19C   XASP20V2   2       15
!
! Volume 3
!
CP        Source     MAINT     394   XASP20V3   3        2
CMS       Source     MAINT     393   XASP20V3   3        3
DUMPVIEW  Source     MAINT     393   XASP20V3   3        4
!
!
/*-------------------------------------------------------------*/
/*          6250 BPI VM/XA NLS Feature Tape                    */
/*-------------------------------------------------------------*/
/*                                                             */
CMS       Base       MAINT     193   R2M0NLS    1        2
```

```
| CP       Object   MAINT    194   R2M0NLS   1        3
| HELP     Files    MAINT    ?     R2M0NLS   1        4
| CMS      Source   MAINT    393   R2M0NLS   1        5
| DUMMY    Object   MAINT    XXX   R2M0NLS   1        6
| GCS      Object   MAINT    595   R2M0NLS   1        7
| /*                                                    */
| /*                                                    */
```

Appendix D. Listing CP Data Areas and Control Blocks

You can obtain a listing of CP data areas and control blocks written in ASSEMBLER by assembling a file called HCPBLOKS ASSEMBLE. (Data areas and control blocks in PLAS are not listed). Follow these steps:

1. Establish the appropriate minidisk access order:

   ```
   vmfsetup 56643089 cp ■
   ```

2. Copy and unpack the HCPBLOKS ASSEMBLE file. You must do this because the UPDATE module requires unpacked files. HCPBLOKS ASSEMBLE is on the CP BASE2 disk (394) in packed format.

   ```
   copy hcpbloks assemble fm = = a (unpack olddate ■
   ```

3. If you have updates to HCPBLOKS ASSEMBLE, use the UPDATE command to apply the updates to the source file:

   ```
   update hcpbloks assemble * hcpxa cntrl * (ctl ■
   ```

   If there are no update files, you will receive the message NO UPDATE FILES WERE FOUND.

4. Issue the GLOBAL command for the macro libraries listed in HCPXA CNTRL:

   ```
   global maclib cpnew hcpgpi hcppsi hcpoml hcpom2 cplib dmsgpi dmsom osmacro ■
   ```

5. Set the virtual punch to remain open after spool files reach 50,000 records:

   ```
   spool punch noeof ■
   ```

6. Assemble HCPBLOKS with HASM. With the options shown below, HASM will generate a PRT file that contains the assembler listing of HCPBLOKS. This listing includes the CP data areas and control blocks.

   ```
   hasm fn (sysparm (exp) xref (full) print ■
   ```

   fn is $HCPBLOK if updates were applied. Otherwise, it is HCPBLOKS.

   ```
   PRT FILE spoolid SENT FROM MAINT PRT AS spoolid RECS nnnn COPY  001 A NOHOLD NOKEEP
   ```

7. You can now erase the work files on your A disk:

   ```
   erase hcpbloks assemble a ■
   erase hcpbloks text a ■
   erase $hcpblok assemble a ■
   erase $hcpblok text a ■
   erase hcpbloks updates a ■
   erase hcpbloks updlog a ■
   ```

Appendix E. Example of Alternate GCS Nucleus Placement

Overview

This appendix demonstrates how to save your GCS system at a virtual storage location other than the one provided by the product tape samples. You might want the GCS segment at a different location, depending on the other segments being used by your GCS group.

This section also describes how to increase the size of the GCS saved system. One reason that you might want to do this is to increase the amount of available common storage. Refer to *VM/XA SP Planning and Administration* for information about calculating common storage requirements for GCS.

To relocate your GCS saved system and change its size, you need to:

- Change the SLC names in the GCSLOAD EXEC
- Create SLC files to contain the new address locations
- Modify the DEFSYS entries in the SAMPNSS EXEC for GCS
- Build and save the GCS nucleus.

Procedure

This sample procedure demonstrates how to relocate your GCS saved system in a 16-megabyte virtual machine. Originally 2M in size and loaded at X'400000', the segment is moved to X'800000' and increased in size to 3MB.

1. Log on as MAINT and access the GCS system disk:

   ```
   access 595 a ■
   ```

2. Modify the GCS loadlist (GCSLOAD EXEC A) and change the Set Location Counter (SLC) values that determine where the shared portion of GCS is loaded into virtual storage.

   In the commands that follow, "SLC L400000" is the name of a CMS file that contains the address of the starting location for loading the main portion of the GCS nucleus. This SLC statement precedes the CSIALP entry in the loadlist. "SLC L600000" marks the end of the GCS nucleus (also the end of common storage), and it precedes the CSIZET entry.

   Note: The GCS loadlist may be changed by service. After applying a PUT, you should check the file and change it if necessary before you rebuild the GCS nucleus.

   Enter the following commands:

   ```
   xedit gcsload EXEC a ■
   set case upper ■
   top ■
   change / SLC L400000 / SLC L800000 /* ■
   top ■
   change / SLC L600000 / SLC LB00000 /* ■
   file ■
   ```

3. Create two new SLC files to match the new loadlist:

   ```
   xedit SLC L800000 a ■
   input $SLC  800000 ■
   set hex on ■
   change /$/X'02'/ ■
   file ■
   xedit SLC LB00000 a ■
   input $SLC  B00000 ■
   set hex on ■
   change /$/X'02'/ ■
   file ■
   ```

   There must be **two** blanks between SLC and the address.

   X'02' is an unprintable loader control character.

   There must be **two** blanks between SLC and the address.

   X'02' is an unprintable loader control character.

4. Modify the DEFSYS entry for your GCS saved system in the SAMPNSS EXEC. As a result, you may also have to change other DEFSYS entries. In this example, the new location for GCS overlays the default locations for CMSVSAM and CMSAMS, so you will have to change those entries.

   Note: To convert an entry in the system name table of a VM/SP or VM/SP HPO system to a DEFSYS entry, see *VM/XA SP Conversion Notebook*.

   The IBM-supplied DEFSYS entry for GCS in the SAMPNSS EXEC looks like this:

   ```
   'CP DEFSYS GCS  0-6 EW 400-5FF SW MINSIZE=256K VMGROUP RSTD'
   ```

   GCS occupies pages 400 – 5FF (all of segments 4 and 5).

   a. Calculate the new values for the DEFSYS entry:

      1) To calculate the page number where the GCS nucleus begins, divide the corresponding SLC value (X'800000' in this example) by the page size, X'1000'. The result is X'800'.

      2) To calculate the page number where GCS ends, divide the corresponding SLC value (X'B00000' in this example) by X'1000' and subtract one. The result is X'AFF'.

   b. XEDIT the SAMPNSS EXEC and enter the new values in the DEFSYS entry for GCS. The changed entry looks like this:

      ```
      'CP DEFSYS GCS  0-6 EW 800-AFF SW MINSIZE=256K VMGROUP RSTD'
      ```

5. Copy the latest version of the product parameter file to the 191 minidisk:

   ```
   copy 56643089 $ppf fm = = a (replace ■
   ```

6. Generate and save the GCS nucleus:

   ```
   access 495 c ■
   ```

   You do not want to change the nucleus on the 595 disk.

7. Before you generate the nucleus, you may want to use GROUP EXEC to create a new GCS configuration file for this nucleus. See the GCS step in the installation procedure.

   ```
   itask build gcs systemname ■
   ipl cms ■
   ```

   If you do not specify a *systemname*, the default is GCS.

Appendix F. Restricted Logon Passwords

The CP nucleus includes a system security feature called ADRP (Auto-Deactivation of Restricted Passwords). ADRP works with a CMS file named RPWLIST DATA that contains the list of restricted logon passwords shown in Figure 68. You can edit this file to add your own restricted passwords.

Issuing the DIRECTXA command causes the system to search the directory for the restricted passwords contained in this list. All passwords that match are changed to NOLOG in the directory before the directory is placed online. You cannot log on to any user ID whose password has been changed to NOLOG. Therefore, make sure that you have changed all restricted passwords in the directory to unique non-restricted passwords before you issue the DIRECTXA command.

```
ACNT     ****************************************************************
AUTOLOG  * THIS MODULE IS RESTRICTED MATERIALS OF IBM.                    *
AUTOLOG1 * 5664-308 (C) COPYRIGHT IBM CORPORATION - 1988                  *
BATCH    * LICENSED MATERIALS - PROPERTY OF IBM                           *
CE       * SEE COPYRIGHT INSTRUCTIONS, G120-2083                          *
CMSUSER  ****************************************************************
CMS1
CMS2     ****************************************************************
CMS3     *                                                               *
CPCMS    *                Restricted Password List                       *
DIRM     *  Format Rules:                                                *
ECMODE   *                                                               *
GCS      *   1) The RPWLIST DATA file must be fixed record length       *
IBMCE    *      format, with a record length of at least 8.              *
IPCS     *   2) Each password must start in column 1.                    *
ISMAINT  *   3) Columns 1-8 must contain restricted passwords only.      *
ITPS     *   4) Each line may contain only one password.                 *
IVPASS   *   5) Column 9 must contain a blank.                           *
LEV2VM   *   6) Columns 10 through 80 may be used for comments, such     *
MAINT    *      as this prologue.                                        *
MASTER   *                                                               *
MDVR     ****************************************************************
OPASS
OPERATNS
OPERATOR
OSVS1
PASSWORD
PRODBM
PROMAIL
PSR
ROUTER
RSCS
SFBATCH
SSFCAL
SQLDBAPW
SQLUSER
SYSADMIN
SYSDUMP
VMAP
VSEIP
VSEIPO
VSEMAINT
```

Figure 68. The RPWLIST DATA File

Appendix G. Controlling Disk String Merges

Automatic Merging

VMFREC normally merges minidisks in accordance with the :MERGE. tag in the product parameter file.

Since disks are merged only within a string (for example, DELTA disks are merged to DELTA disks, not to APPLY disks), merging does not consolidate the contents of all disks, but saves back levels of service on the current disks and frees the alternate disks for new levels of service.

Preventing Automatic Merging

You may not want VMFREC to merge certain disk strings. A common reason for suppressing the merge is to receive multiple COR tapes on the alternate DELTA and APPLY disks so that a single APPLY step can be performed. In the more general case, you may wish to group the contents of service tapes differently than would result from using the default :MERGE. tags. Always merge the APPLY and DELTA strings the same way because they work in pairs.

To prevent VMFREC from merging, choose one of the following methods:

- Remove the symbolic disk string names from the :MERGE. tag in the PPF. Do not remove the merge tag itself because it is a required tag.
- Define and format an additional minidisk and put it at the head of the string that you do not want to merge. VMFREC will not merge if the top disk is empty. The method will require you to include the new top disk as the leftmost disk in the string in the :MDA. section of the PPF.

Remember to make changes to the PPF with overrides. Also remember that if you are processing COR service, you must have an additional COR override. Below is an example of an override file called NOMERG $PPF that will suppress all merging for the CMS component.

```
:OVERLIST. CMS CORCMS
*
:CMS. CMS 56643089
:MERGE.
:END.
*
:CORCMS. CMS 56643089
:MERGE.
:END.
```

Manual Merging

Merging One Disk to Another

To merge a disk to another disk, you must first copy the entire contents of the origin disk to the destination disk and then erase the entire contents of the origin disk.

The filemode number must be included on the COPYFILE commands because if a file of the same filename and filetype exists on the destination disk with a different filemode number, the filemode number of the destination disk is retained. Since the filemode numbers are used to indicate the applied status of update files and $PTFPART lists, it is very important to retain the filemode number from the origin disk.

After all of the files on the origin disk are merged (that is, copied) to the destination disk, use the erase option of the access command to clear the disk. You must then create a dummy file on the origin disk in order to commit the erase:

```
access origin_addr origin_fm ■
access destination_addr destination_fm ■
copy * * origin_fm0 = = destination_fm0 (olddate replace ■
⋮
copy * * origin_fm6 = = destination_fm6 (olddate replace ■
access origin_addr origin_fm (erase ■
xedit dummy file origin_fm ■
  input dummy ■
  file ■
erase dummy file origin_fm ■
```

Merging an Entire Disk String

To merge a disk string that contains 3 layers (production, intermediate, alternate), you must use the above procedures to merge the intermediate to the production and then merge the alternate to the intermediate.

Appendix H. Service Reference Tables

VM/XA SP Filetypes and Abbreviations

The first column of Table 22 shows the 3-character abbreviations conventionally used for the filenames in the second column. These abbreviations are used in the filetype of parts supplied for service (see "Parts Supplied for Service and What to Do with Them" on page 865).

Table 22 (Page 1 of 2). VM/XA SP Filetypes and Abbreviations

Abbreviated Filetype	Complete Filetype
$CN	$CONSTS
$EX	$EXEC
$PF	$PPF
$XE	$XEDIT
ASM	ASSEMBLE
AXA	AUXXA
AXM	AUXMXA
CON	CONSTS
CPY	COPY
CTL	CNTRL
DAT	DATA
DLB	DOSLIB
DLK	DOSLNK
DRC	DIRECT
DSF	DSF
EAM	EXCAMENG
EUC	EXCUCENG
EXC	EXEC
EXL	EXECLIST
HAB	HELPABBR
HCM	HELPCMS
HCP	HELPCP
HCQ	HELPCPQU
HCS	HELPCPSE
HDI	HELPDEFI
HDS	HELPDISP
HDU	HELPDUMP
HED	HELPEDIT

Table 22 (Page 1 of 2). VM/XA SP Filetypes and Abbreviations

Abbreviated Filetype	Complete Filetype
HEP	HELP
HEX	HELPEXEC
HE2	HELPEXC2
HGR	HELPGROU
HHE	HELPHELP
HIN	HELPINDI
HLP	HLP
HMA	HELPMACR
HME	HELPMENU
HMO	HELPMONI
HMQ	HELPCMSQ
HMS	HELPMSG
HPF	HELPPF
HPR	HELPPREF
HPU	HELPPURG
HQU	HELPQUER
HRX	HELPREXX
HSE	HELPSET
HSR	HELPSRPI
HSS	HELPCMSS
HST	HELPSTOR
HTR	HELPTRAC
HTS	HELPTASK
HWT	HCPSADWT
HXE	HELPXEDI
LAN	LANGUAGE
LDM	LOADMAP
LDR	LOADER

Table 22 (Page 2 of 2). VM/XA SP Filetypes and Abbreviations	
Abbreviated Filetype	**Complete Filetype**
LEF	LEF0000
LKC	LKEDCTRL
LKE	LKEDIT
LOL	LOADLIB
L00	L000000
L03	L00E000
L20	L020000
L23	L023000
L40	L400000
L50	L500000
MAC	MACRO
MAP	MAP
MEM	MEMO
MLB	MACLIB
MOD	MODULE
PRF	PROFILE
PRS	PRODUCTS
REP	REPOS
SAE	SAMPEXEC
SAP	SAMPLIST
SCR	SCRIPT
SCU	SCRIPTUC
SOR	SOURCE
SYN	SYNONYM
TLB	TXTLIB
TXT	TEXT
XAM	XAMAINT
XED	XEDIT

Parts Supplied for Service and What to Do with Them

Table 23 shows the parts that IBM will supply for service to each part of your system. The first column identifies the part to be serviced by filetype. The second column shows whether that kind of part is serviced by update, replacement, or both. The third and fourth columns list the parts that IBM will send you when you apply COR service and PUT service, respectively. The fifth column shows the parts required for update service: the source file (for example, an ASSEMBLE file), which you already have, the update, which is supplied on the COR or PUT tape, and the auxiliary control file, which is generated during the service process. The sixth column shows the most important EXECs and commands you will use in rebuilding the serviced part. (For full instructions, see Chapter 14, "Rebuilding CMS after Applying Service" on page 431, Chapter 15, "Rebuilding CP after Applying Service" on page 487, Chapter 16, "Service to DV" on page 507, or Chapter 17, "Service to GCS" on page 517.) If the sixth column is blank, the part supplied by IBM simply replaces the one you already have. If you want to keep your own part for reference, you can rename it.

Table 23 (Page 1 of 2). Parts Supplied for Service and What to Do with Them

Usable Form	Update or Replace?	Parts for COR Service	Parts for PUT Service	Update Parts	Build Function
EXEC	Update $EXEC	Update AUXXA EXC*nnnnn* EXEC $EX*nnnnn*[1] $EXEC[1]	Update AUXXA EXC*nnnnn* EXEC $EX*nnnnn*[1] $EXEC[1]	Update AUX*xxxxx* $EXEC	EXECUPDT Copy *fn* EXC*nnnnn fm* = EXEC =[2]
	Replace EXEC	EXC*nnnnn* EXEC	EXC*nnnnn* EXEC		Copy *fn* EXC*nnnnn fm* = EXEC =
XEDIT	Update $XEDIT	Update AUXXA XED*nnnnn* XEDIT $XE*nnnnn*[1] $XEDIT[1]	Update AUXXA XED*nnnnn* XEDIT $XE*nnnnn*[1] $XEDIT[1]	Update AUX*xxxxx* $XEDIT	EXECUPDT Copy *fn* XED*nnnnn fm* = XEDIT =[2]
	Replace XEDIT	XED*nnnnn* XEDIT	XED*nnnnn* XEDIT		Copy *fn* XED*nnnnn fm* = XEDIT =
MACLIB	Update MACRO	Update AUXXA MAC*nnnnn*[1] MACRO[1]	Update AUXXA MAC*nnnnn*[1] MACRO[1]	Update AUX*xxxxx* MACRO	VMFMAC
	Replace MACLIB		MACLIB		
COPY	Update COPY	Update AUXXA CPY*nnnnn*[1] COPY[1]	Update AUXXA CPY*nnnnn*[1] COPY[1]	Update AUX*xxxxx* COPY	VMFMAC

Table 23 (Page 2 of 2). Parts Supplied for Service and What to Do with Them

Usable Form	Update or Replace?	Parts for COR Service	Parts for PUT Service	Update Parts	Build Function
MODULE	Update ASSEMBLE	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1]	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1] MODULE[1]	Update AUX*xxxxx* ASSEMBLE	CMSGEND UTILITY GENMOD
	Replace text deck	TXT*nnnnn*	TXT*nnnnn* MODULE		Copy *fn* TXT*nnnnn fm* = TEXT =
	Replace MODULE	MOD*nnnnn* MODULE	MOD*nnnnn* MODULE		
NUCLEUS	Update ASSEMBLE	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1]	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1]	Update AUX*xxxxx* ASSEMBLE	VMFBLD HCPLDR
	Replace text	TXT*nnnnn*	TXT*nnnnn*		VMFBLD HCPLDR
TEXT	Update ASSEMBLE	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1]	Update TXT*nnnnn* ASM*nnnnn*[1] ASSEMBLE[1]	Update AUX*xxxxx* ASSEMBLE	VMFHASM Copy *fn* TXT*nnnnn fm* = TEXT =[2]
	Replace text deck	TXT*nnnnn*	TXT*nnnnn*		Copy *fn* TXT*nnnnn fm* = TEXT =
$PPF	Replace $PPF	$PF*nnnnn* $PPF	$PF*nnnnn* $PPF		Copy *fn* $PF*nnnnn fm* = $PPF =
Other parts	Replace part	*aaannnnn* real filetype	*aaannnnn* real filetype		Copy *fn* *aaannnnn ft* = *ft* =

nnnnn The number of the most recently applied PTF.

xxxxx Is 1- to 5-alphanumeric characters.

aaa The text deck filetype prefix from the AUX file identification record in the main control file.

Notes:

[1] This is a base part. This part is shipped from IBM only when it is a new part or when the part (source) is replaced.

[2] This procedure is used when there are no local modifications.

Appendix I. How to Find the PTF Number from the APAR Number

If you know the APAR number associated with an update, you can find the corresponding PTF number. For example, suppose that you want to find the PTF number for APAR 01010.

1. Find the filename of a module affected by the APAR. (If you are looking for the PTF number associated with an update listed in a DEPEND entry, you already know one.)

```
listfile * *01010* * ■
HCPABC   A01010HP fm
⋮
```

2. List all text decks with the filename of that module:

```
listfile hcpabc txt* * (EXEC ■
```

The **EXEC** option saves the list in a file called CMS EXEC. Any previously existing CMS EXEC on your A-disk is erased.

3. Look at each text deck until you find one that lists the APAR you are looking for:

```
xedit hcpabc txtnnnnn ■
locate /01010 ■
```

```
A01010HP  COR  UM00000  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A34567HP  COR  UM00001  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
A56789HP  COR  UM00002  * Comments
* PREREQ: NONE
* CO-REQ: NONE
* IF-REQ: NONE
* DEPEND: A01010HP
A88888HP  COR  UM00003  * Comments
⋮
(Executable text is here.)
```

The APAR number is the 5 numeric characters in the first word of the text deck line. The corresponding PTF number is the third word of the same line.

Appendix J. Messages

This appendix contains messages issued by the service EXECs. For other messages, see the *VM/XA SP System Messages and Codes Reference.*

002E **[Input|Overlay] {File[(s)] |Dataset|Note} [*fn* [*ft* [*fm*|*dirname*]]] not found**

Explanation: The specified file was not found on the accessed disks. Either the file does not reside on this filemode, the file identification was misspelled, or incomplete identification was provided to cause the appropriate filemode to be searched, or system disk was not accessed as a read-only extension of filemode A.

For the PRELOAD command, either the loadlist EXEC, the CNTRL file, or one of the input text files could not be found.

For SETPRT command, the module represented by *fn ft* does not exist in the current CMS search order.

For the STATEW command, the file may exist, but it is not on any of the user's read/write filemodes.

For the ZAP command, either none of the libraries specified for a TXTLIB or LOADLIB could be found, or the INPUT filename could not be located via the STATE macro.

For the ZAPTEXT and EXPAND commands, the input text file or INPUT filename could not be located via the ESTATE command.

For the VMFLKED command, either you specified a file that cannot be found on a filemode in the CMS search hierarchy, or you specified a filename on a %CONTROL statement as the name of a CNTRL file and that file was not found.

For the VMFPLCD EXEC, the specified file was not found. If the specified file is the envelope file it must exist for any functions except DUMP, WGS, or RST. If the file ID is not the envelope file, then it was a file specified on a SCAN or SKIP file which could not be found within the constraints of the option EOG or EOD limit.

For the CONVERT command, the input DLCS file you specified was not found.

See the *CMS Command Reference* for a description of the file identification required by each command and the search procedure used.

System Action: RC=20|28|36.
Execution of the command is terminated. The system status remains the same.

For DMSSPR, nothing has been sent to the virtual 3800.

For DMSLIO, some loader information fields have been initialized, but they should not interfere with a subsequent LOAD command.

For the CONVERT command, conversion stops. RC=44.

For the VMFPLC2 command, the STOP option has been specified with the LOAD function, and the file was not found in alphabetic sequence. The tape is positioned immediately before the next file.

For VMFPLCD, if RC=20, the envelope file ID is not a valid CMS file ID. If RC=36, the disk was not accessed at all, or not accessed in R/W mode. For the VMFLKED and VMFZAP commands, processing ends.

For the VMFMERGE command, other required files are checked and then processing ends.

User Response: Find or create the desired file. To make sure the file exists, issue LISTFILE fn ft * or STATE fn ft *. Correct and reissue the command.

For DMSSPR, access the disk having the required module or respecify a different module in the calling sequence and then reissue the SETPRT command.

For a DMSROS TEXT file, ensure that the file is accessible and reissue the command.

For the VMFLKED command, make certain that the proper disks are accessed and check the name of the specified file. If the name was specified incorrectly, reissue the command with the correct name. For VMFPLCD, correct the file ID, reposition the envelope file if necessary and reenter the command.

For VMFTXT:

If the filetype is EXEC, make sure that a memberlist EXEC file exists and that the filename of the memberlist and the libname parameter are

spelled the same. Correct the error and reissue the command.

If the filetype is CNTRL, make sure that the specified CNTRL file exists and is correctly spelled. Correct the error and reissue the command.

If the filename and filetype pair is one of the following:

VMFMSGS EXEC
VMFDATE MODULE
VMFTXT DATA

contact your system programmer and arrange to have these files installed again on the CMS system disk as filemode 2 files.

For the VMFZAP, VMFMERGE, and VMFREMOV commands, see if the proper disks are specified in the VMFPARM file and then re-issue the command.

For the CONVERT command, correct the filename or access a disk or directory where the file can be found.

003E Invalid option *option*.

Explanation: The specified option is invalid. It may have been misspelled, or, if the option is truncatable, it may have been truncated improperly, or it may conflict with another option in the command line.

System Action: RC=8.

Execution of the command is terminated. The system status remains the same.

For DMSLIO, some option processing may have caused such things to happen as user storage to be cleared or the location counter to be set. This should not interfere with a subsequent LOAD command.

For the VMFLKED command, processing ends.

User Response: Correct and reenter the command.

010S Premature end occurred on *fn ft fm*

Explanation: The physical limits (forward or backward) of the envelope file were exceeded or a command completed before the group count specified by option EOG was satisfied.

System Action: Processing is terminated. RC=40.

User Response: Use positioning commands to reset the position within the envelope and reenter the command. Use an EOG count (or option EOD) to limit the command if necessary.

040E No file loaded [for *fn ft*]

Explanation: The user has not previously issued a LOADMOD or LOAD command, or the module consists of zeros.

System Action: Execution of the command is terminated. The system status remains the same. The position of the envelope file (if any) depends on the limits specified by option EOG or EOD. RC=40.

User Response: Load files via the LOAD or LOADMOD command. If using envelope files, reposition the envelope file and reenter the command with a different file ID to be loaded.

048E Invalid {filemode|output filemode} [*mode*]

Explanation: This message can occur for any one of the following reasons:

- The filemode was not specified correctly.
- For most CMS commands, filemode 'S' is an invalid mode.
- For the DLBL command, filemode 'R' and 'T' should not be used.
- The filemode number, if specified, is not between 0 and 6.
- More than 2 characters were specified for the filemode.
- A null line was entered as the first specification with the MULT option of the DLBL command.
- The filemode specified with a LISTDS command was not the mode of an OS or DOS disk.
- The filemode specified with a LISTFILE command was not the mode of a CMS-formatted disk.
- For XEDIT, if a filemode number is not specified, a 1 may be appended to the invalid filemode.
- For VMFPLCD, the filemode specified on a VMFPLCD LOAD command to which the files are to be loaded is either invalid or was specified as '*'.

System Action: Execution of the command is terminated. The system status remains the same. The position in an existing envelope, if any, is not affected. RC=24.

For DMSTPI or VMFPLC2, if the DEN, 9TRACK, or 18TRACK options were specified, the modeset byte has been set for the specified device (TAPi, where *i* = 1, 2, 3, 4) or TAP1.

User Response: Reissue the command with the filemode specified correctly.

050E Parameter missing after *value*.

Explanation: A parameter that is required by the command was not specified.

For the ASSGN command, the disk mode must be specified for the SYS*aaa* logical unit.

For the DLBL command, the disk mode or DUMMY or CLEAR must be specified after the *ddname*.

For the FILEDEF command, the device name or DUMMY or CLEAR must be specified after the *ddname*.

For the NUCXDROP command, a required parameter that must follow a function is missing.

For the SET command, a required parameter that must follow a function is missing.

For the XMITMSG command, one of the options required a value to follow it, but the end of the parameter list was reached.

System Action: RC=8.

Execution of the command is terminated. The system status remains the same.

User Response: Correct and reissue the command.

056E File *filename filetype filemode* contains invalid record format.

Explanation: For DMSLBM and DMSNCP (GEN, ADD, REP), the specified file is not in the expected format. MACRO and MEND cards must be included in the MACRO files, and the prototype card must be specified with a name that does not exceed 8 characters. If an &COPY statement appears, it must contain a name. A MACLIB must contain "LIB" in columns 4−6 of record one.

For DMSLBT, the specified file has more than 255 entry points (ESD only), or has records which are incompatible or missing. The NAME field in the CSECT instruction of the specified file must have a valid symbol or label.

For DMSLIO, an invalid condition was found in a TEXT or TEXTLIB file. TXTLIB files created on EDF disks must have "PDS" in columns 4−6 of record one. TXTLIB files created on non-EDF disks must have "LIB" in columns 4−6 of record one. RLD data must be compatible with the TEXT file or TXTLIB member to which is belongs. If an ICS statement was submitted, the specified name was previously defined, or the initial length of the CSECT was not found in the ESD card.

For DMSSYN, the specified file is not in the expected format. The SYNONYM file must contain 80-byte records in free form format, with columns 73−80 ignored. The data consists of a command name followed by a blank and the user synonym. This may optionally be followed by a count which is preceded by at least one blank.

For DMSZAP, either the header record for TXTLIB or LOADLIB was invalid, or the pointer to the directory or module map was in error.

For the VMFTXT command, the memberlist EXEC file was not in the required format.

System Action: RC=100.
Execution of the command is terminated. The system status remains the same. For DMSGLB, the library is not globally searched, and the operation continues for any other libraries named in the command.

For the VMFTXT command, the invalid record is ignored. Processing continues for any remaining records in the file.

User Response: For DMSLBM and DMSNCP, issue the MACLIB COMP command, then check the MACLIB with a MACLIB MAP command. Correct the format error.

For DMSGLB, the specified library does not have "LIB" in columns 1−3 or 4−6 of the first record. One possible cause is the library may be in packed format. Correct the library and reissue the command.

For DMSSYN, correct the format of the file.

For DMSLIO, recreate the TXTLIB or TEXT file.

For DMSLBT, if the message specifies ESD, check for more than 255 entry points for a member; otherwise, check for invalid or missing records. If the NAME field in the CSECT instruction was left blank, enter a valid symbol or label.

For the VMFTXT command, correct the invalid entry in the memberlist EXEC file. If the member specified in the invalid record has a filetype of TEXT, you may issue the:

```
TXTLIB VMFTXT ADD membername
<(FILename<)>>
RENAME VMFTXT TXTLIB A libname
TXTLIB A
```

commands. If the filetype is not TEXT, then erase VMFTXT TXTLIB A and then reissue the command.

For DMSZAP, recreate the library or module.

Then reissue the command.

062E **Invalid character** *char* **in** *filename filetype filemode.*

Explanation: The character specified was invalid in the file ID in which it appeared.

System Action: RC=100.

Execution of the command is terminated. The system status remains the same.

User Response: Reissue the command, specifying the correct file ID.

065E *option* **parameter specified more than once.**

Explanation: The option was specified more than once in the command line.

System Action: RC=100.

Execution of the command is terminated. The system status remains the same.

User Response: Reissue the command, specifying the option only once.

066E **The variations of this message are explained below.**

- *option1* **and** *option2* **are conflicting options**

  Explanation: The specified options are mutually exclusive and must not be in the same command.

  System Action: Execution of the command is terminated. The system status remains the same. RC=24.

  User Response: Correct and reissue the command.

- *option1, option2, option3, option4, option5* **are conflicting options**

  Explanation: The identified options are mutually exclusive and must not be specified in the same command.

  System Action: Processing is terminated. The position in an existing envelope, if any, is not affected. RC=24.

  User Response: Reenter the command with only one of the options.

104S **Error** *nn* **reading file** *fn ft fm* **[from {disk|XEDIT}] [RC=***rc* **from** *command*]

Explanation: An irrecoverable error occurred while reading the file from disk. *nn* indicates the nature of the error; it may be one of the following:

Code	Meaning
1	The specified file was not found.
2	The buffer area is not within user storage limits.
3	A permanent disk read error occurred. This may occur if you link to and access another user's disk, and try to read a file that was refiled by its owner after you issued the ACCESS command. Reissue the ACCESS command and try to read the file again.
5	The number of records is less than zero or more than 32,768.
7	The fixed/variable flag in the file status table entry is not F or V.
8	The given storage area was smaller than the actual size of the records read. (This error is valid if reading the first portion of a large record into a small buffer. It does not cause the function to terminate.)
9	The file is open for writing and must be closed before it can be read.
11	Only one record can be read for a variable-length file. In this case, the number of records is greater than 1.

12 An unexpected end of file occurred (the record number specified exceeds the number of records in the file).

13 A variable-length file has an incorrect displacement in the active file table.

14 An invalid character was detected in the filename.

15 An invalid character was detected in the filetype.

19 An I/O error occurred on a FBA device. This was indicated by a non-zero condition code from a DIAGNOSE code X'20'. Error detected in a module DMSDIO.

25 Insufficient virtual storage is available.

26 Requested item number is negative, or item number plus number of items exceeds file system capacity.

27 An attempt was made to update a variable-length item with one of a different length.

System Action: Execution halts. The system remains in the same status as before the command was entered.

For DMSEDI, the edit session is terminated. If the error occurred during a RENUM operation, the workfile is erased and the file being edited remains unchanged.

For DMSEXL, the file specified in the EXECLOAD command was not loaded into storage. The execution of the command is terminated.

For DMSDSL, the condition of the DOSLIB file is unpredictable.

For DMSGLO, no global variable table(s) were created.

For DMSLBM, the condition of the MACLIB file is upredictable.

For DMSGLB, that library is not globaled, but the operation continues for any other libraries named in the command.

For DMSXGT, if the error occurred during a GET operation, the subcommand is terminated and the editing session continues.

For DMSXIN, the execution of the command or subcommand is terminated. If multiple files were being edited, the editing session continues for those files.

For DMSXRE, if the error occurred during a RENUM operation, the subcommand is terminated and the editing session continues.

For the TAPE command, if the DEN, TRTCH, 7TRACK, 9TRACK, or 18TRACK options were specified, the modeset byte has been set for the specified device (TAPn, where n is a character from 0 to 9 or A to F; the default is TAP1). Some records may have been written on tape. RC=100 or RC=1*nn* (*nn* described above).

User Response: If you can determine the problem from the "Explanation" and remedy the condition, reissue the command. If not, retry the command, and if the problem persists, call your system support personnel.

For DMSDGL, verify DOSLIB integrity with the DOSLIB MAP command.

For DMSGLB, RDBUF has returned an RC other than 0, 1, or 8. RC=1 indicates it is an OS/DOS DISK and RC=8 occurs if the LRECL is greater than 80. Either of these conditions is acceptable.

For DMSLBM, verify MACLIB integrity wiht the MACLIB MAP command.

For DMSLIO and DMSMOD, reissue the entire LOAD/INCLUDE sequence after checking the error conditions.

The problem may be that the in-core directory for the M-disk that contains the file being loaded, does not match the actual directory. The real disk directory may have been changed since the disk was last accessed, or if on the system disk, the saved system may need resaving.

For error code '09', issue an FSCLOSE macro for the file. If a permanent disk read error occurs (code 3), it may be the result of the user having detached a virtual disk without releasing it. CMS, not realizing that the disk is no longer a part of the virtual machine, assumes that the disk is still active and encounters an error when it tries to read or write the file.

For VMFPLCD, if processing is not started from the beginning, care should be taken to reposition the envelope record pointer using the positioning commands, as the resulting state of the record position is not predictable after this error.

105S **Error *nn* writing file *fn ft fm* [{on disk|to XEDIT}] [RC=*rc* from *command*]**

Explanation: An irrecoverable error occurred while writing on disk. nn indicates the nature of the error; it may be one of the following:

Code	Meaning
2	The virtual storage address is 0.
4	The first character mode is invalid.
5	The second character mode is invalid.
6	The number of records in the file is too large; it cannot exceed 65,533.
7	An attempt has been made to skip over an unwritten variable-length item.
8	The number of bytes was not specified.
9	The file is already active for reading.
10	The maximum number of CMS files (3,400) has been reached.
11	The fixed/variable flag is not F or V.
12	The disk is not a CMS read/write disk.
13	The disk is full (recoverable error).
14	The number of bytes to be written is not integrally divisible by the number of records to be written.
15	The length of this record is not the same as that of the previous record.
16	The fixed/variable flag is not the same as that of the previous record.
17	A variable-length record is greater than 65K bytes.
18	The number of records is greater than 1 for variable-length file.
19	The maximum number of data blocks per file (16,060) has been reached.
20	An invalid character has been detected in filename.
21	An invalid character has been detected in filetype.
22	Virtual storage capacity has been exceeded.
25	Insufficient virtual storage is available.
26	Requested item number is negative, or item number plus number of items exceeds file system capacity.
27	An attempt was made to update a variable length item with one of a different length.

System Action: Execution of the command terminates. The system status remains the same.

For DMSDSK, the reader file is saved. The status of the output file is unpredictable.

For DMSEDI, the edit session terminates. The status of the file is as it was before the edit session or at the execution of the last SAVE subcommand or automatic save. The RENUM workfile is erased. A workfile, EDIT CMSUTI, may have been created on the input disk.

For DMSEXL, the file specified in the EXECLOAD command was not loaded into storage. The execution of the command is terminated.

For DMSDSL, the condition of the DOSLIB file is unpredictable.

For DMSLBM, the condition of the MACLIB file is unpredictable.

For DMSLBT, DMSLST, DMSMOD, DMSSRT, DMSTPD, and DMSUPD, the status of the output file is unpredictable.

For DMSRDC, the reader is closed with a HOLD status to preserve the file. However, if *nn*=13 and the error occurs while writing the last block of the file to disk, then the file will have already been purged before the reader is closed.

For DMSTPI, if the DEN, TRTCH, 7TRACK, 9TRACK, or 18TRACK options were specified, the modeset byte has been set for the specified device (TAPn, where *n* is a character from 0 to 9 or A to F; the default is TAP1). The status of the output file is unpredictable. The tape may not be in the same position as before the command was entered.

For DMSXCP, the EXCP request fails with the return code nn. Check the attributes of the file specified in the DTF and DLBL.

For DMSXFD, if the error occurred during a FILE or SAVE, a temporary work file, XEDTEMP CMSUT1, may have been created on the input disk.

For DMSXPT, if the error occurred during a PUT (D) operation, the subcommand is terminated and the editing session continues.

For DMSXRE, if the error occurred during a RENUM operation, the subcommand is terminated and the editing session continues. RC=100.

User Response: If you can determine the problem from the "Explanation" above and remedy the condition, reissue the command. If not, reissue the command, and if the problem persists, call your system support personnel.

For DMSDSL, use the DOSLIB MAP function to verify DOSLIB integrity.

For DMSLBM, use the MACLIB MAP function to verify MACLIB integrity.

For DMSLIO, reissue the LOAD/INCLUDE sequence from the beginning, after checking the above error conditions.

For DMSXCP, specify a smaller partition with the SET DOSPART command, or use the CP define storage command for a larger machine and IPL CMS.

For VMFPLCD, before reentering the command, it may be necessary to reposition the envelope file.

814E Formats for the message follow:

- **Message number *nnnn*, format *nn*, line *nn* was not found; it was called from *routine* in application *applid*.**

  Explanation: The message requested could not be found in the specified repository.

  System Action: RC=12.

  User Response: Verify the command and reissue it.

- **{Message number|Dictionary item} *nnnn*, format *nn* not found**

  Explanation: VMFPLC or VMFPLCD issued a request to VMFPLCM to issue a message or to retrieve a dictionary item. The message number or the dictionary item number is not supported by VMFPLCM.

  System Action: If a message number was not found, the action is dependant upon the error detected by the program requesting the message to be issued. For dictionary items, the resulting output which was to use the dictionary item will be incorrect or untranslated. RC=12.

  User Response: This is an internal error to the VMFPLC EXEC or VMFPLCD EXEC. The problem should be reported to your IBM Service Representative for correction.

961E There are class *class* files in the *device*

Explanation: A virtual device (RDR, PRT, or PUN) has been checked and found to have files of the indicated class when none were expected.

System Action: Processing end, RC=100.

User Response: Change the class of the files in the device, move them, or purge them. Run the program again.

1447E Invalid command format

Explanation: Either the VMFPLC ROUTE command was issued with TAPE routing specified and there were parameters beyond the TAPE parameter, or the VMFPLC ROUTE command was issued with DISK routing specified and a partial envelope file ID was provided. For DISK routing, the envelope file ID, if provided, must specify the *fn ft fm*.

System Action: No action taken. RC=24.

User Response: Reenter the command in the correct format.

1448E Options *option... option* invalid for *function*

Explanation: The identified options are invalid for the function requested.

System Action: Processing is terminated. The position in an existing envelope, if any, is not affected. RC=24.

User Response: Reenter the command with a valid combination of functions and options.

1450S Output file *fn ft fm* disk is read-only

Explanation: Attempted to write to the identified file, which is on a read-only disk.

System Action: Processing is terminated. RC=36.

User Response: Correct the disk access to read/write and reenter the command. It may be necessary to reposition the envelope file prior to reentering the command.

1451S Cannot dump an envelope file to itself. File *fn ft fm* is the same envelope file.

Explanation: Attempted to dump the identified file to an envelope file and the file is either the envelope itself or a copy of the envelope file under a different file ID.

System Action: Processing is terminated. RC=32.

User Response: Correct the command so that it does not dump the identified file. Prior to reentering the command, the envelope record position should be reset to a valid point for continuation using the various positioning commands.

1452S Unidentifiable envelope control record in *fn ft fm*

Explanation: A record was encountered in the envelope file that has a prefix which identifies it as a VMFPLCD control record but which is neither a group separator record nor a file header record.

System Action: Processing is terminated. RC=100.

User Response: Reposition the envelope file using the positioning commands and reenter the failing command. If the failure persists, contact your IBM Service Representative for assistance.

1453S Loading of file *fn ft fm* would overlay envelope.

Explanation: During a LOAD operation, a file was found which, if loaded, would overlay the envelope file (this may occur, for example, if a file that is being loaded has the same file ID as the envelope file).

System Action: Processing is terminated and the position within the envelope file is left at the file that caused the failure. RC=32.

User Response: Rename the envelope file or change the filemode to which the files from the envelope are to be loaded. If the envelope file is renamed, it will be necessary to reposition the envelope using the new name before proceeding.

1454S Date/Time stamp update on *fn ft fm* failed.

Explanation: After a file was loaded, an attempt to update its date/time stamp to the value of the file when it was dumped failed.

System Action: Processing is terminated and the position within the envelope file is left at the file that caused the failure. RC=104.

User Response: Ensure that the DMSPLU MODULE used to update the date/time stamp is available. Reposition the envelope file if required and reenter the command.

1455S Error occurred trying to truncate file *fn ft fm*

Explanation: A DUMP or WGS command was issued while the envelope file was positioned at other than the end of the file. As a result, an attempt was made to truncate the existing envelope prior to the DUMP or WGS. An error was detected while truncating the file.

System Action: Processing is terminated and the status of the envelope file is unpredictable. RC=100.

User Response: The envelope file will need to be rebuilt from the start.

1456S Unexpected error *rexx code* from VMFPLCD.

Explanation: Syntax error in the VMFPLCD EXEC was found by REXX. The REXX code identifies the type of error.

System Action: Processing is terminated. The results of the error are unpredictable. RC=104.

User Response: This may be an internal logic error in the VMFPLCD EXEC. Contact your IBM Service Representative for help.

1457E Envelope file was not specified.

Explanation: The VMFPLCD command did not specify the file ID of the envelope file and no GLOBALV variable exists that specifies the file ID.

System Action: Processing is terminated. RC=24.

User Response: Reenter the command specifying the file ID of the envelope file to be processed or created.

1458E **File designated as envelope on *function* is not an envelope**

Explanation: The first record in the file designated as the envelope is not a VMFPLCD generated control record; thus, the file is not an envelope file.

System Action: Processing is terminated. RC=32.

User Response: Correct the file ID on the command to point to an envelope file.

1459S **Number of records for file *fn ft fm* different than when dumped.**

Explanation: A file was found during a LOAD operation that contains a different number of records than when it was originally dumped.

System Action: Processing is terminated and the position within the envelope is left at the start of the file causing the error. RC=40.

User Response: Reposition the envelope and retry the command. If the failure persists, the assistance of your IBM Service Representative may be required to determine the cause.

1460E **Envelope file *fn ft* exists but is on R/O extension of the *fm* disk.**

Explanation: The envelope file specified on a DUMP or WGS function does not exist on the requested filemode, but a file with the same filename and type exists on a read-only extension of that filemode.

System Action: Processing is terminated. The position in an existing envelope file, if any, is not affected. RC=36.

User Response: If the envelope file ID is correct, either do not make the other mode a read-only extension, or rename the file on that extension before reentering the command. If the file ID was not the one desired, reenter the command with a new envelope file ID.

1801E **There is no tape mounted on 181. Mount the correct tape and restart the receive procedure.**

or

An envelope formatted file cannot be located.

Explanation:

Version 1: VMFREC expects a tape to be mounted on 181 and there was not.

Version 2: The envelope file following the ENV keyword could not be located on any accessed disk.

System Action: VMFREC exits with RC=100.

User Response:

Version 1: Mount tape to be received and restart VMFREC.

Version 2: Verify the file specification and location of the envelope file. Reenter the command with the correct parameters.

1802R **The {tape mounted|envelope} is at a lower service level than the existing service map.**
The service {tape|envelope} is at service level *level1*.
The SERVICE DISKMAP file indicates service level *level2*.
Is this what you want? (YES/NO)

Explanation: VMFREC will receive data from a PUT that is at a lower level than was previously received.

System Action: If response to prompt is no, then VMFREC exits. Otherwise, receive procedure continues.

User Response: Respond to prompt, either YES or NO.

1803E **The tape is in the wrong format. Mount the correct tape and restart the receive procedure.**

or

The envelope is in the wrong format. Check the syntax and reenter the command with valid envelope parameters.

Explanation:

Version 1: VMFREC expects the first tape file to contain a file that determines the tape type. This file is either incorrect or not present.

Version 2: VMFREC uses the first file within the envelope file to determine envelope type such as COR or PUT. This file is incorrect or not present. The envelope type cannot be determined.

System Action: VMFREC exits with RC=100.

User Response:

Version 1: Mount the correct tape.

Version 2: Reenter the command with the name of a valid envelope file following the word ENV.

1804I **Receiving service for [component *comp* of] product *prodid***

Explanation: This is an informational message indicating which component or product service is being received for.

System Action: VMFREC continues receiving service.

User Response: None.

1805E **[No|Not enough] parameters were entered. Check the syntax and reenter the command.**

Explanation: An attempt was made to execute a procedure, but the required parameters were not entered.

System Action: The procedure exits with RC=8.

User Response: Review the documentation and restart the procedure with correct parameters.

1806I **The current SERVICE DISKMAP file contains the map of the {mounted tape|current envelope}.**

Explanation: An informational message indicating that the tape currently mounted or the envelope being used has been previously mapped.

System Action: VMFREC continues receiving service.

User Response: None.

1807E *routine* **cannot continue; the** *filename filetype* **file was not found.**

Explanation: A procedure requires a file that cannot be found on an accessed minidisk.

System Action: The procedure exits with RC=28.

User Response: The file named in the message needs to be made available. Verify that the minidisk access is as it should be. Check the product parameter file to verify that the correct minidisk is listed in the component minidisk assignment (MDA) section.

1808E **The** *ppfname* **$PPF product parameter file was not found.**

Explanation: A procedure requires a product parameter file in order to set required parameters, and the file cannot be found on an accessed minidisk.

System Action: The procedure exits with RC=28.

User Response: The file named in the message needs to be made available. Verify that the minidisk accesses are as they should be.

1809E **A** *function* **{tape|disk} error has occurred with return code** *rc.*

Explanation: VMFREC has encountered a serious tape or disk error.

System Action: The procedure exits with the return code received from the VMFPLC2 command.

User Response: Correct the tape or disk drive problem as indicated by the *rc* from the VMFPLC2 command. If drive is working then the tape or disk might not be in the correct format for the procedure.

1810R **Enter a component name or type QUIT.**

Explanation: The procedure needs a component name in order to carry out its function.

System Action: The procedure waits for a valid component name to be entered or QUIT.

User Response: Enter a component name listed in the product parameter file for the product requested.

1811W **The access of** *vdev* **failed for product** *prodid.*

Explanation: An access command has failed for the minidisk listed in the message text.

System Action: The procedure exits with RC=4.

User Response: Verify that the minidisk is linked correctly.

1812E **Service for *prodid* is not {on|in} the {mounted tape|current envelope} but the *prodid* product ID is in the SERVICE DISKMAP file. It is {on|in} the service {tape|envelope} mapped with a relative {tape|envelope} number of *number*.**

Explanation: The wrong tape volume is mounted on tape 181, or the wrong envelope has been specified.

System Action: The procedure exits with RC=28.

User Response: Mount the correct tape volume or specify the correct envelope and restart the procedure.

1813E ***option1* and *option2* are conflicting options. Check the syntax and reenter the command.**

Explanation: Conflicting options have been entered.

System Action: The procedure exits with RC=8.

User Response: Check the documentation, then restart the procedure using correct command syntax.

1814E **Service for the product ID *prodid* is {on|in} the {mounted tape|current envelope}, but the *prodid* product ID is not in the SERVICE DISKMAP file.**

Explanation: The tape or envelope has not been mapped correctly.

System Action: VMFREC exits with RC=28.

User Response: Erase existing service map and restart VMFREC.

1815E ***parameter* is an unknown parameter. Check the syntax and reenter the command.**

Explanation: An unknown parameter was detected by the procedure.

System Action: The procedure exits with RC=8.

User Response: Check the documentation, then restart the procedure using correct command syntax.

1816E ***parameter* is an unknown parameter. This product ID is not {on the tape|in the envelope} or in the SERVICE DISKMAP file.**

Explanation: The first or only parameter entered is invalid.

System Action: The procedure exits with RC=8.

User Response: Check that the product ID that was entered is valid.

1817E **The *updateid-name* update ID is not in the *filename* CNTRL file.**

Explanation: The update ID entered on the command line, or listed in the product parameter file, was not found in the control file listed in the product parameter file.

System Action: The procedure exits with RC=28.

User Response: Verify that the correct update ID was entered, either on the command line or in the product parameter file, according to the procedure syntax. Check that the correct control file is listed in the component section of the product parameter file.

1818E **The product service header file cannot be found.**
The {tape|envelope} positioning might be in error.
Restart the receive procedure beginning with service for product ID *prodid*.

Explanation: VMFREC can not find the product header file on the tape in the tape file calculated, or in the envelope at the file location calculated.

System Action: VMFREC exits with RC=28.

User Response: The tape or envelope positioning might be incorrect upon return from an unsupported product service EXEC. If this is thought to be the case, restart the procedure beginning with the product listed in the message text. If this is not the case, the file counts in the component list of the product parameter file might be incorrect.

1819E **The control file *filename filetype* was not found.**

Explanation: The control file listed in the component section of the product parameter file cannot be found.

System Action: The procedure exits with RC=28.

User Response: Verify that the control file listed is correct, and that the minidisks are accessed correctly.

1820E **The build list *filename filetype* was not found.**

Explanation: The build list listed in the component section of the product parameter file cannot be found.

System Action: The procedure exits with RC=28.

User Response: Verify that the build list listed is correct and that the minidisks are accessed correctly.

1821E **All filemodes are being used. The previous access order will be restored.**

Explanation: Access to required minidisks failed because no filemodes were available.

System Action: The procedure exits with RC=100.

User Response: Release unwanted minidisks. Reduce the number of minidisks listed, as required in the component section of the product parameter file.

1822W **Access to *vdev* cannot be restored.**

Explanation: Upon exit, the procedure could not reaccess the minidisk listed.

System Action: The procedure continues with RC=4.

User Response: Upon exit, reaccess the minidisk if needed.

1823W **The *filename filetype* PTF parts list file was not found. A PTF might be missing for product ID *prodid*.**

Explanation: A file listed in a PTF parts list could not be found.

System Action: The procedure continues with RC=4.

User Response: Investigate why the PTF part is missing. Check the minidisk assignments (MDA) section for the component in the product parameter file.

1824E ***vdev*, accessed as *filemode*, is not R/W.**

Explanation: The procedure requires the minidisk listed in the message text to be read/write.

System Action: The procedure exits with RC=100.

User Response: Reaccess the minidisk as read/write, and restart the procedure.

1825W **A part handler cannot be found for part *part-type* while applying PTF *ptfnum* to product ID *prodid*.**

Explanation: A part is listed in the component section of the product parameter file with a part-specific EXEC that cannot be found on an accessed minidisk.

System Action: The procedure continues with RC=4.

User Response: If this was a part that was intentionally bypassed, then no action is required. Otherwise, find the part listed in the component section of the product parameter file and verify the listed part handler spelling. Check the minidisk accesses for correctness.

1826W **The part type *filename filetype* was not found while executing the *name* part handler.**

Explanation: A part listed in the component section of the product parameter file cannot be found on an accessed minidisk.

System Action: The procedure continues with RC=4.

User Response: Find the part listed in the component section of the product parameter file and verify that the part is spelled correctly. Verify the minidisks are accessed correctly.

1827S **The *name* part handler failed for part *fn ft* while applying PTF *ptfnum* for product ID *prodid*.**

Explanation: The part handler listed in the message text failed.

System Action: The procedure exits with RC=100.

User Response: Determine the cause of the part handler failure by examining its error messages.

1828W **The prefix *xxx* is listed in the control file *filename*, but a PTF number cannot be found.**

Explanation: A prefix (*xxx*) has been listed on a control file level indicating that the text deck will be named by a PTF number, but an auxiliary control file containing a valid PTF number cannot be found.

System Action: RC=4.
Text deck is named according to the control file without the use of a PTF number.

User Response: Check the control file and auxiliary control structure.

1829W **The *filename filetype* update cannot be applied.**

Explanation: The VMFAPTXT part handler failed while attempting to apply the update listed in the message text.

System Action: The procedure continues with RC=4.

User Response: Examine previous messages to determine the cause.

1830E ***label* has been specified as the target for the component *compname* of product *prodid*. There is no target listed on the *label* tag.**

Explanation: A symbolic string has been specified as a target, but the label in the minidisk assignments (MDA) section of the $PPF file does not contain any minidisk address.

System Action: The procedure exits with RC=100.

User Response: Correct the MDA section in the file and reinvoke the procedure.

1831W **PTF *ptfnum* is included in *filename filetype*. It is in the exclude list *filename2 filetype2*, but cannot be excluded.**

Explanation: The PTF specified in the $EXCLIST is contained in a part of the PTF being applied. VMFAPPLY can only exclude PTFs serially from a text deck (the last PTF applied).

System Action: The procedure continues with RC=0.

User Response: Refer to *VM/XA SP Installation and Service* for instruction on how to process PTFs that could not be excluded by VMFAPPLY.

1832W **Text deck *filename filetype* is included in the *name* build list but cannot be found.**

Explanation: A text deck, listed in the build list to be included in the build, cannot be found on an accessed minidisk.

System Action: The procedure continues with RC=4.

User Response: Check that the correct minidisks are listed in the component section of the product parameter file. Verify that this deck was intentionally removed from the access order.

1833W **Text deck *filename filetype* does not contain any executable code but is listed in the *name* build list. The build process might not be complete.**

or

Text deck *filename filetype* has been selected as the highest level deck according to the AUX file structure but it does not contain any updates. The build process might not be complete.

Explanation:

Version 1: A text shell (a text deck that contains only requisite information) has been found. It is listed in the prologue of another text deck as an update.

Version 2: While examining the control file to locate the latest text deck an AUX or update file was located. This indicates the deck identified in the message as the latest. The existence of the AUX or update file also indicates that the deck has been updated but the self-documenting prolog does not contain any update or AUX file entries. Entries

may have been added to the control or AUX file but not assembled into the deck.

System Action: The procedure continues with RC=4.

User Response:

Version 1: Investigate why this text deck is missing.

Version 2: Examine the control file and associated AUX files to verify that all updates have been correctly assembled into the textdeck. Examine the text deck prolog to ensure the update entries are in the correct format.

1834W **The filetype of *filename filetype* might be incorrect due to preferred AUX files.**

Explanation: The filetype of a text deck was found on a level of the control file that also contains preferred files.

System Action: The procedure continues with RC=4.

User Response: Check the control file named in the component section of the product parameter file for correctness.

1835E **AUX file *filename auxft* on *vdev* could not be saved by XEDIT when adding update *filename update-ft*. PTF(s) included in *filename* might be partially applied.**

Explanation: VMFAPPLY could not completely process all the changes included in the text deck that is a part of the PTF being applied.

System Action: The procedure exits with RC=100.

User Response: Add another APPLY minidisk to the $PPF file or increase the size of the current APPLY minidisk and restart VMFAPPLY.

1836E **This program requires [filemode A|*mode*] to be accessed as R/W. [Access mode A as R/W and re-try.]**

Explanation: The procedure cannot continue without the indicated read/write filemode.

System Action: The procedure exits with RC=100.

User Response: Access a read/write filemode A and restart the procedure.

1837W ***filename filetype filemode* update entries do not match the control file for update *filename filetype*.**

Explanation: While checking the entries in the text deck with the AUX entries in the AUX file and update entries in the control file, a mismatch has been detected.

System Action: The procedure continues with RC=4.

User Response: Investigate the problem with textdecks not matching the AUX file and control file.

1838E **The service exec requires filemode C to be accessed as R/W. Access mode C as R/W and retry.**

Explanation: For a product with its own service EXEC, a read/write filemode C must be accessed.

System Action: The procedure exits with RC=100.

User Response: Access a read/write filemode C and either copy the existing SERVICE DISKMAP for filemode A to filemode C, or erase any on filemode A before reinvoking VMFREC.

1839W **Multiple update files for product ID *prodid* have been found on: *vdev***

Explanation: Duplicate update files exist on different minidisks for the same product.

System Action: The procedure continues with RC=4.

User Response: Be aware that, with multiple updates, there is a possibility that the wrong decks can be picked up at build time.

1840W **The AUX file entries in *filename1 filetype1 filemode1* and *filename2 filetype2 filemode2* do not match.**

or

***filename filetype filemode* contains AUX entries but an AUX file has not been located.**

Explanation: While comparing AUX entries between the given text deck and AUX file or between the two given AUX files, a mismatch has been detected. The PTF that was being processed

when the error was detected may be partially applied.

System Action: The procedure continues with RC=4.

User Response: Investigate the reason for the mismatch, correct it, and retry the procedure.

1841W **The *filename filetype* update file was not found.**
[An update shell has been created.]

Explanation: An update file pointed to by an entry in a text deck prologue or an AUX file could not be found on an accessed minidisk. Update shells are only created for text decks. When the update file is for another type of part (for example, a MACRO or an EXEC), the second part of this message is not issued and no update shell is created.

System Action: The procedure continues with RC=4.

User Response: Investigate why the update file listed in the message text was not found.

1842W **The *filename filetype* file was not found. The exclude option was ON but there is no exclude list. The process is continuing with the exclude option set OFF.**

Explanation: VMFAPPLY cannot locate the $EXCLIST file for the control file specified in the product parameter file.

System Action: The procedure continues with RC=4.

User Response: Investigate why the $EXCLIST file listed in the message text was not found.

1843W **The *filename filetype* update cannot be applied for PTF *ptfname*. *vdev* must be R/W.**

Explanation: The update files are marked "applied" by renaming from filemode number 1 to 5. The disk that the files reside on must be read/write in order to rename.

System Action: The procedure continues with RC=4.

User Response: If the update file should have been applied, then move the file to a read/write minidisk and restart the procedure.

1844W **For product *prodid*, PTF *ptfnum* part *fn ft* was not found.**

Explanation: A part listed in a PTF part list could not be found.

System Action: The procedure continues with RC=4.

User Response: Investigate why the part was not found.

1845W ***filename filetype* file is a text deck shell for PTF *ptfnum*.**

Explanation: A text deck that is listed as a part of the PTF listed in the message text could not be found.

System Action: The procedure continues with RC=0.

User Response: Investigate why the deck was not found. This situation can occur when applying more than one PTF which affects the same part. If a text deck is shipped for the part and the part is source-maintained, only the text deck for the latest PTF is sent. When this text deck is received during VMFREC, text deck shells are created for any included PTF for which a full text deck does not exist. Only the latest text deck is sent for source-maintained parts because all other levels of the text deck can be obtained by assembling the source with the appropriate updates applied.

During VMFAPPLY, this message is issued whenever a text deck listed as part of a PTF is a text deck shell rather than a full text deck. Normally, these messages will be removed from the APPLY error log when a PTF is applied for which a full text deck exists. However, if apply processing stops before such a PTF is applied these messages will remain in the error log. You should determine why apply processing has stopped and run VMFAPPLY again.

1846W **Update file *fn ft* has a requisite of *ptfnum*. The update file for part *part-type* cannot be found on a R/W disk. The DEPEND entry cannot be entered.**

Explanation: A text deck contains a requisite that points to an update file that cannot be found, or a depend entry cannot be written because the update file is on a R/O disk. The depend function is bypassed.

System Action: The procedure continues with RC=4.

User Response: Investigate why the requisite update file was not found.

1847W *ptfnum1* **is referenced in update file** *fn ft* **and is a part of PTF** *ptfnum2*. *ptfnum1* **is a requisite for another component.**

Explanation: This is a warning message, indicating that a requisite exists that is outside of the component.

System Action: The procedure continues with RC=0.

User Response: None.

1848E *filename ptf-ft* **on** *vdev* **could not be copied/renamed to** *base-ft* **on** *vdev*. **AUX files and update files show all service in text deck** *filename ptf-ft* **has been applied.**

Explanation: VMFAPPLY could not complete processing a text deck that is part of the PTF. The text deck being processed needs to be renamed to its base filetype, and copied to the minidisk identified in the message. This text deck is not supported by VMFBLD.

System Action: The procedure exits with RC=100.

User Response: Copy and rename the text deck identified in the message to another minidisk listed for the component in the $PPF file, or increase the size of the current DELTA minidisk and copy/rename the text deck.

1849W **APAR** *aparnum1* **is a requisite of APAR** *aparnum2* **and the aux entry** *update-file-type* **is not included in the self documenting prolog information of the text deck** *filename filetype*.

or

APAR *aparnum* **is a required requisite, but part** *filename filetype* **on disk** *address* **has not been applied by VMFAPPLY and the update file is not fm5.**

Explanation:

Version 1: An update file was found for the text deck listed in the message and this update file was not included in the self documenting prolog information of that text deck.

Version 2: An update file was found for the APAR listed in the message and this update file was not applied by VMFAPPLY.

System Action: The procedure continues with RC=4.

User Response:

Version 1: Investigate why the aux entry identified in the message is missing from the updated text deck. The text deck may have been built without a requisite APAR.

If this text deck was built by IBM, this text deck is in error. Report this to your IBM service representative.

If the text deck was built locally, you may have left out a required entry in an aux file for the text deck identified in the message. Correct the aux file and run VMFBLD again.

Version 2: Investigate why this update file has not been applied.

1850I **The processing for** *part-name* **by the** *filename* **EXEC was bypassed.**

Explanation: Processing of a part has been bypassed. This could occur if the part handler has been commented out in the product parameter file.

System Action: The procedure continues.

User Response: If the part was expected to be handled, investigate why the part handler EXEC was not found.

1851I **Processing** *filename filetype* **with the part handler** *parthandler* **EXEC.**

Explanation: This is an informational message indicating that the function requested is progressing to the next part.

System Action: The procedure continues.

User Response: None.

1852I **This is *volume-number* of *total-volumes*, level *level* {PUT|COR} {tape|disk}.**

Explanation: An informational message indicating the level of the tape or disk being processed.

System Action: The procedure continues.

User Response: If you are using the correct tape or envelope level, then no response is needed. Otherwise, stop the procedure, mount the correct tape or use the correct envelope, and restart procedure.

1853I **Processing PTF *ptfnum*.**

Explanation: This is an informational message indicating that the function requested is progressing to the next PTF listed in the PTF parts file.

System Action: The procedure continues.

User Response: None.

1854W ***ptfnum* is a requisite for another component.**

Explanation: This message indicates that a PTF has been found that lists a requisite that is not a part of the component being processed.

System Action: The procedure continues with RC=4.

User Response: Verify that the out-of-component requisite will be applied.

1855W **Control file *fn* contains invalid data. Text deck *fn* will not be named by PTF number.**

Explanation: A record was found in the control file that violates the control file rules as defined in *VM/XA SP CMS Command Reference* under the UPDATE command.

System Action: RC=4.
The text deck is named according to the control file without the use of a PTF number.

User Response: Check the control file and AUX structure for invalid level IDs or update identifiers.

1856I **PTF *ptfnum* is in the *filename filetype* exclude list for product ID *prodid* , and will not be applied.**

Explanation: This is an informational message indicating that a PTF listed in the apply list will not get applied, because it is also listed in the exclude list.

System Action: The procedure continues.

User Response: None.

1857I **APAR *aparnum* is a required requisite, but part *filename1 filetype1* on disk *disk address* has not been applied by VMFAPPLY and the update file is not fm5.**

Explanation: A textdeck includes a requisite which has an update that has not been applied as indicated by a filemode other than *fm1*.

System Action: The procedure continues with RC=0.

User Response: The message is informational. The part identified in the message text is a part that must be processed manually. VMFAPPLY changes the filemode of update files for TEXT decks to a 5 when processed. Update files for other parts, such as EXECs, XEDIT macros, MACROs and COPY files, will force this message to be issued since filemodes on update files for these parts are 1 (VMFAPPLY does not process these update files). The message can be eliminated by changing the filemode on update files for these other parts to a 5. Next time VMFAPPLY is run (you do not have to run VMFAPPLY again now), these messages will not be issued.

1858I ***message***

Explanation: The variations of this message are explained below.

System Action: In each case, the procedure continues.

User Response: In each case, refer to *VM/XA SP Service Guide* for instructions on how to regenerate service parts of the system.

Messages:

- *filename* **build list contains files that have been serviced.**

  Explanation: VMFAPPLY has processed a PTF that contained a new level of a file contained in the build list found in the build (BLD) section of the product parameter file. VMFBLD should be invoked to regenerate the object associated with the build list, such as the nucleus and the module.

- *filename filetype* **has been serviced.**

  Explanation: VMFAPPLY has processed a PTF that contained a new level of the file identified in the message. A build step might be required to make the change available for execution.

1859I **The *name* build exec completed with return code *rc*.**

Explanation: The part handler identified has completed the build step required for a build list specified in the product parameter file.

System Action: The procedure continues, with RC=*rc*.

User Response: None.

1860E ***compname* is an invalid component name for product ID *prodid*.**

Explanation: The base *compname* listed on the override compname tag was not found in the complist tag.

System Action: The procedure exits with RC=100.

User Response: Verify that the override and complist tags match. If not, change the tag that is incorrect.

1860I ***compname* is an invalid component name for product ID *prodid*.**

Explanation: A component name has been entered for a product that does not contain that component.

System Action: The procedure continues and prompts you for a valid *compname*.

User Response: Respond to the prompt by entering a valid *compname*.

1861W **The *filename* service exec completed with return code *rc message*.**

Explanation: The service EXEC has completed the task. The return code (*rc*) indicates whether the exec was successful or not at completing the task. *message* may be one of the following according to the return code (*rc*) specified:

- *rc*=0

  The service has been received.

  VMFREC has invoked a product service EXEC, and the EXEC has completed successfully.

- 0 < *rc* < 4

  The service received might have been done previously.

  VMFREC has invoked a product service EXEC, and the EXEC has completed successfully. Service for this EXEC may have already been received before.

- 3 < *rc* < 8

  Service received, a build might be required.

  VMFREC has invoked a product service EXEC, and the EXEC has completed successfully. The product for which service was received may need to be rebuilt in order to apply the service.

  User Response: Review service procedures for the product service to check if a build is required.

- 7 < *rc* < 12

  Receive has been discontinued, some service may have been received.

  VMFREC has invoked a product service EXEC and the EXEC was halted before service was completely received.

  User Response: Correct the problem that caused the product service EXEC to fail and restart VMFREC.

- *rc* > 11

  Receive has been discontinued, tape position unknown.

  VMFREC has invoked a product service EXEC and the EXEC has been halted at an unknown point in the process.

User Response: Correct the problem that caused the product service EXEC to fail and restart VMFREC.

System Action: VMFREC exits with *rc* unless the list option was specified, in which case the next product in the list is processed.

User Response: None, unless otherwise indicated above.

1862I **The LTO option was specified, but not all targets are accessed R/W as required. The LTO option has been set OFF.**

Explanation: In order to specify the last-text-only option, the target minidisks must be accessed as read/write.

System Action: The procedure continues with the last-text-only option turned off.

User Response: If the last-text-only option is desired, stop the procedure and reaccess the target minidisks in write mode, then restart the procedure.

1863W **Service has been received on the S minidisk. Module generation and/or reIPL of CMS might be required.**

Explanation: Service has been received onto the system minidisk.

System Action: RC=4.

User Response: The system minidisk needs to be re-IPLed, or the CMS segment might need to be resaved.

1864E *Service exec* **previously failed for** *prodid compname1* **on PUT** *level1*. **You must rerun** *service-exec* **for** *prodid compname1* **on PUT** *level1* **before running** *service-exec* **for** *prodid compname2* **on PUT** *level2*.

Explanation: More than one $ER* erase list has been found in the access order. Update files from the level 1 PUT were not processed completely.

The variables in the message text are defined as follows:

service-exec	The main EXEC name (VMFREC)
prodid	The product ID
compname1	The previously failing component ID
level1	The failing PUT level
compname2	The current component ID
level2	The current PUT level.

System Action: RC=23 (for merge error). The function is terminated.

User Response: The function cannot be reinvoked until one of the duplicate $ER* erase lists is removed from the access order. The component identified in the message text should be run after the duplicate file is removed.

1865W **Update files for** *prodid compname* **have been found on the target. The target should be empty. The MERGE tag parameter in the $PPF file might be improperly specified.**

Explanation: The target minidisk should be empty at the start of receive for update files. If the target minidisk is listed on the MERGE tag in the PPF correctly, the minidisk should be empty.

System Action: RC=4. A prompt allows the user to continue with the process or discontinue receive.

User Response: Unless the intent is to receive updates to a minidisk that has already received updates, the user should exit and check the PPF for correct string labels on the MERGE tag.

1866W **Duplicate update files have been found in the target and APPLY access orders. Merge might not have been done correctly.**

Explanation: Duplicate update files have been found on the target and APPLY minidisk strings.

System Action: Update file processing continues with RC=4.

User Response: With the proper use of the service tools and MERGE tag in the PPF, this situation should not occur. Manual file manipulation is necessary in order to remove the duplicate files.

1867E **The loadlist** *loadlist-name* **EXEC** [*vdev*|*dirname*] **does not contain a valid loader name.**

Explanation: The loader invoked for the build of the component being processed is determined by the loader card specified in the loadlist identified in the :BLD. section of that component in the product

parameter file. This loadlist does not contain a loader card that is recognized by the EXEC as a valid loader.

System Action: The procedure exits with RC=100.

User Response: Examine the loadlist for a loader card. If one does not exist then add one. If one does exist then it is not supported by the parthandler invoked to process it. Modify the product parameter file to invoke a parthandler that does support the format of this loadlist.

1868W **Auxfile** *filename1 filetype1* **contains PTF#** *ptfnumber* **but textdeck** *filename2 filetype2* **could not be located. Search will continue with the level ID from the control file.**

or

The control file *filename* **CNTRL indicates that part** *filename filetype* **should exist but it can not be located. Searching will continue at the next level.**

Explanation:

Version 1: In determining the filetype of a textdeck using the AUX and control file structure, an AUXFILE has been found that contains a valid PTF or local tracking number. A textdeck filetype formed with this number and prefix cannot be located using the current access order. This may result in a build being done with a lower level textdeck.

Version 2: In determining the filetype of a textdeck using the AUX and control file structure, an update or AUX file was specified in the control file, and the update or AUX file exists on an accessed disk but a part with a filetype of the prefix concatenated with the LEVELID could not be located.

This condition may also exist if the update specified in the control file contains a prefix but the next AUX level points to an AUXFILE that does not contain a valid PTF number. In this case, a search is made using the LEVELID from the control file which may cause the message to be issued as described above.

System Action: In each case, the procedure will continue to process the control file searching for a textdeck filetype. RC=4.

User Response: In each case, investigate the access order to determine if the textdeck identified in the message can be located. If it can be located, either the disk should be added in the PPF, or the textdeck copied to a disk that is listed in the PPF. If the deck cannot be located, then the module should be reassembled.

1869R *prodid compname* **begins on PUT|COR** *tapenum* **volume** *vol.* **Mount volume** *vol* **and press ENTER or type QUIT.**

Explanation: The service you are receiving for a product/component begins on another tape.

System Action: The procedure continues when you press ENTER, or exits if you type QUIT.

User Response: Mount the tape containing the service for the component and press ENTER. Alternatively, type QUIT, mount the next tape and reinvoke VMFREC to load the rest of the service for the component.

1870R **Choose the component which is to be processed from the list below:**

Explanation: A procedure has been requested, but a component name was not specified. The names displayed are taken from the COMPLST and OVERLSTP tags in the product parameter file.

System Action: The system waits for a response to the prompt.

User Response: Enter a component name from the list displayed, or enter QUIT.

1871R **Do you want to continue? (YES/NO).**

Explanation: An opportunity is given to prematurely exit the procedure.

System Action: The system waits for a response to the prompt.

User Response: Answer the prompt.

1872E **COR is a nonsupported option for the** *prodid* **product ID. The {tape|envelope} has been positioned to the beginning of the product's first file. Use the product's corrective service procedure to process.**

Explanation: The product specified on the VMFREC command is not packaged in the format required by VMFREC. Refer to the product's documentation for procedure to follow for corrective service.

System Action: The procedure exits with RC=8.

User Response: None.

1873W *prodid compname* **relative {tape|envelope} file** *number, label* **is listed in the product directory but is not in the $PPF file. This {tape|envelope} file will not be received.**

Explanation: The product specified on the VMFREC command is not packaged in the format required by VMFREC. Refer to the product's documentation for procedure to follow for corrective service.

System Action: The procedure continues with RC=4.

User Response: Correct the receive service (RECSER) section in the $PPF file for the component specified on the VMFREC command. Add the label and part handler for the tape or envelope file.

1874R **The** *execname* **option {PUT|COR} does not match the type of the {mounted tape|current envelope} {PUT|COR}. Do you want to continue? (YES/NO)**

Explanation: VMFREC is receiving service from a preventive or corrective tape or envelope but the option specified on the commmand does not match the tape or envelope type.

System Action: The procedure exits if response is NO, continues if response is YES.

User Response: Enter YES or NO.

1875E **XEDIT cannot save file** *filename filetype* **on disk** *filemode.* **The return code from XEDIT was** *rc.* **Correct the problem and re-start the procedure from the beginning.**

Explanation: The procedure being run could not save the file being edited. The minidisk might be full.

System Action: The procedure exits with RC=100.

User Response: Look up the return code from XEDIT and determine the cause of failure. Correct the problem and restart the procedure from the beginning.

1876E **The VMFAPPLY option PUT|COR does not match the apply list specified.**

Explanation: VMFAPPLY was invoked with the PUT or COR option, but the $APPLIST being used does not match the option specified.

System Action: The procedure exits with RC=100.

User Response: Reinvoke VMFAPPLY, using the correct option.

1877R *prodid compname* **begins on PUT|COR** *tapenum* **volume** *vol.* **Do you want to continue? (YES/NO)**

Explanation: VMFREC is receiving service for a component that started on a previous tape. You should not receive this service if you have not already received the service from the other tape.

System Action: The procedure exits if response is NO, continues if response is YES.

User Response: Enter YES or NO.

1878I **No files have been found for** *prodid compname* **on the PUT|COR level** *level.*

Explanation: VMFREC could not find any service for the product/component on the tape that is currently mounted.

System Action: The procedure continues with RC=0.

User Response: Check other tape volumes for this service level for the product/component.

1879R *product-name component-name* **on PUT|COR** *level-number* **volume** *vol-number* **continues on volume** *vol-number2.* **Mount volume** *vol-number2* **and press ENTER or type QUIT.**

Explanation: The service you are receiving for a product/component is continued on another tape.

System Action: The procedure continues when you press ENTER, or exits if you type QUIT.

User Response: Mount the tape containing the remainder of the service for the component and press ENTER. Alternatively, type QUIT, mount the next tape and reinvoke VMFREC to load the rest of the service for the component.

1880W *filename filetype* **is an update file that has a requisite for APAR** *aparnum*, **but no update file can be found.**

Explanation: VMFAPPLY found a requisite APAR for the update being applied, and could not locate an update file that contained the APAR number in the filetype. The filetype of the update file being applied could not be added to the DEPEND tag for this update.

System Action: The procedure continues with RC=4.

User Response: Locate the update files for the APAR, and add the filetype specified in the message to the DEPEND tag. The situation where no update files can be found is not an error condition if the PTF that corresponds to the given *ft* does not contain any parts that are serviced with update files.

1881W **Text deck** *filename filetype* **on** *vdev* **contains an applied update for APAR** *aparnum*, **but the AUX file cannot be found on the APPLY string. The PTF was not applied.**

Explanation: The text deck being processed contains an update that has a filemode number of 5, which indicates it has been added to an AUX file. VMFAPPLY could not find an AUX file on any minidisk listed on the APPLY tag in the $PPF file.

System Action: The procedure continues with RC=28.

User Response: Change the filemode number of the update file to a 1 and reinvoke VMFAPPLY, or create an AUX file for the text deck on a minidisk listed in the APPLY string, and add the update file.

1882W **AUX file** *filename filetype* **on** *vdev* **is inconsistent with the text deck** *filename filetype* **on** *vdev*. **The PTF was not applied.**

Explanation: The text deck being processed contains an update that has been applied, but it is not contained as the first entry in any AUX file found in the APPLY string. VMFAPPLY cannot determine where the update being applied should be added to the AUX file.

System Action: The procedure continues with RC=4.

User Response: Change the filemode number of the update file to a 1 and reinvoke VMFAPPLY, or add the update to an AUX file for the text deck on a minidisk listed in the APPLY string.

1883W **There was a problem loading the apply and exclude lists.**

Explanation: VMFREC was not able to copy/rename the apply and exclude contained on the service tape to the target minidisk. The minidisk might be full, or an incorrect level of the apply or exclude list might have been shipped on the tape.

System Action: The procedure continues with RC=28.

User Response: Check the target minidisk for the correct level of the apply and exclude list. If found, then copy/rename them to $APPLIST and $EXCLIST. If they are not found, contact your IBM support center.

1884W *ptfnum* **is an included PTF in** *filename1 filetype1*. **It has been added to** *filename2 filetype2* **apply list.**

Explanation: While applying a PTF, a text deck or AUX file was processed that contained additional PTFs that were not in the $APPLIST. VMFAPPLY will process these PTFs as if they were in the $APPLIST, to ensure that all service required for the PTF is applied.

System Action: The procedure continues with RC=0.

User Response: None.

1885W *filename1 filetype1* **listed in** *filename2* **CNTRL represents service at a higher level than** *filename1 filetype3*. **The higher-level service may need to be reworked or removed.**

Explanation: VMFAPPLY found an AUX file or update file at a higher level in the control file than the AUX file being updated by VMFAPPLY. If the file found is an AUX file, its contents follow the message.

System Action: The procedure continues with RC=0.

User Response: The higher-level service must be investigated to determine if it conflicts with the new

service in the AUX file being updated by VMFAPPLY. Any service that is duplicated by the new service must be removed from the higher level. Any service that conflicts with the new service must be reworked so that it no longer conflicts.

1886W **Update file *filename filetype* on *addr* has a filemode other than fm5 but auxfile *filename filetype* on *addr* contains an AUX entry for this update. The PTF was not applied because of this inconsistency.**

Explanation: During apply processing of a text deck, the deck is read from bottom up looking for AUX entries. When an AUX entry is found the update filetype is taken from the first field and a check has been made of the update filemode in order to determine if the update has been applied. The update has been found with a filemode other than 5 indicating it is not applied. The AUX file was then checked and an entry for this update has been located indicating the update has been previously applied.

System Action: Processing by the current parthandler will end resulting in the PTF not being applied. RC=4.

User Response: Investigate if duplicate updates exist that have already been applied, indicated by having a filemode 5. If so, the current update can be renamed to fm5 or erased. If a duplicate is not found then the update should be renamed to fm5. VMFAPPLY can then be used to reapply this PTF or it can be applied manually.

This message may indicate that a PTF has incorrectly been unapplied. It also may indicate that update files have been received by a part handler other than VMFRCUPD.

1887W **The *filename* EXEC file was not found. The {user exit|part handler} *filename* was not executed.**

Explanation: The specified EXEC was not found.

System Action: The procedure continues without executing the EXEC. RC=4.

User Response: If the EXEC was needed for processing, ensure that the EXEC is available to be executed, and restart the procedure.

1888W **Product parameter file *prodid* $PPF has been found on filemode *filemode*. It has been renamed to *prodid* $PPFSAVE *filemode*. A new product parameter file will be loaded from the {tape|envelope}. Merge any local changes from the renamed file to the new file.**

Explanation: If you had a copy of the product parameter file on the target disk for VMFREC, it would be overlaid. A copy is saved as filetype $PPFSAVE and a new $PPF is loaded down to the target disk for VMFREC.

System Action: The procedure continues with RC=4.

User Response: Merge your changes from the $PPFSAVE file into the new product parameter file loaded down to the target disk.

1889E **The *prodid* $PPF product parameter file contains invalid data on the *ppftag* record. Correct the file and retry.**

Explanation: An error has occurred because of invalid data contained on the product parameter file record indicated by the message text.

System Action: The procedure exits with RC=100.

User Response: Correct the invalid data, and restart the procedure.

1890E **The *ppfname* $PPF product parameter file is invalid. *record* is missing or out of order. Correct the file and retry.**

Explanation: A required record cannot be found in the product parameter file.

System Action: The procedure exits with RC=100.

User Response: Add the required record, and restart the procedure.

1891E **Part handler *name* failed for component *compname* of product ID *prodid*.**

Explanation: A part handler EXEC that is listed in the product parameter file for the component indicated by the message text has failed.

System Action: The VMFBLD procedure exits with RC=100. VMFREC will continue with any other *prodid* components, if the list option was specified.

User Response: Correct the problem that is causing the part-specific EXEC to fail, and retry the procedure.

1892E **The *name* build list was not found in the *ppfname* $PPF file.**

Explanation: A build list name is missing.

System Action: The procedure exits with RC=100.

User Response: Add a build list name to the PPF.

1893E **The *name* tag found in apparent $PPF override file *ppfname*.**

Explanation: A tag was found in a $PPF override file that was not expected. The most likely possibility is that it is a COMPLST tag.

System Action: The procedure exits with RC=24.

User Response: Correct the $PPF file and restart the procedure.

1894W **Ensure that all parts of *prodid compname* have been received before proceeding to the APPLY step.**

Explanation: In order to do a complete APPLY, it is necessary that the entire component be received. This message is issued if part of the component has not been received during this invocation of VMFREC.

System Action: The procedure continues, but will exit with RC=4.

User Response: Receive the rest of the component, if you have not already done so.

1895E ***name1* tag found {before|after *name2* tag in *ppfname* $PPF product parameter file.**

Explanation: A tag was not in the proper place in the $PPF file.

System Action: The procedure exits with RC=100.

User Response: Correct the $PPF and restart the procedure.

1896E **The component name is missing from the component override tag *name*.**

Explanation: An override was specified, but no base component was listed in the override $PPF file.

System Action: The procedure exits with RC=100.

User Response: Correct the $PPF file, and restart the procedure.

1897E **Error reading file *filename filetype filemode*.**

or

A problem occurred reading file *filename filetype* while function *function* was executing.

Explanation: A file required by the procedure cannot be read.

System Action: The procedure exits with RC=100.

User Response: Correct the problem with the file and retry the procedure. Some possible problems may be: not enough storage, or bad file pointers.

1898E **A record can not be written to file *filename filetype filemode*.**

Explanation: A problem has been encountered in trying to add a record to a file.

System Action: The procedure exits with RC=100.

User Response: Correct the problem with the minidisk and retry the procedure. The most likely cause of this problem is that the minidisk is full.

1899E **There is a problem loading a file.**

Explanation: A problem has been encountered writing a file.

System Action: The procedure exits with RC=100.

User Response: Correct the problem with the minidisk and retry the procedure. The most likely cause of this problem is that the minidisk is full.

1900W **The existing *ppfname* $SETUP A1 file has been refreshed. You might want to check your access order when done.**

Explanation: VMFSETUP has been invoked for accessing two consecutive times, without being invoked with the RESTORE option. The first access order has been lost.

System Action: The procedure continues with RC=4.

User Response: Check your access order at the completion of processing.

1901E **VMFSETUP, when invoked with the PPFTEMP parameter, expects a temporary product parameter file. *ppfname* $PPFTEMP was not found.**

Explanation: VMFSETUP was invoked with the PPFTEMP operand. This operand is only to be used if VMFOVER has been invoked previously to create a $PPFTEMP file. Because no $PPFTEMP file was found, VMFOVER could not have run successfully.

System Action: The procedure exits with RC=28.

User Response: Invoke VMFOVER before executing VMFSETUP, or invoke VMFSETUP without the PPFTEMP operand.

1902E **The *compname* component name was not found in the *ppfname* $PPFTEMP file.**

Explanation: The $PPF file does not contain a section for the specified component.

System Action: The procedure exits with RC=100.

User Response: Correct the $PPF file, and restart the procedure.

1904E **The current access order does not match the access order that is in the RELEASE section of *ppfname* $SETUP. Your access order will be left as it is.**

Explanation: VMFSETUP cannot restore the original access order, because the current access order is not the same as the one set up by VMFSETUP when it was originally invoked.

System Action: The current access order is left untouched, and the procedure exits with RC=100.

User Response: Check your access order.

1905W **The access of *vdev* failed with a return code of *rc*. Processing continues for product *prodid*.**

Explanation: VMFSETUP was not able to perform an access that was listed in the $PPF file. The minidisk may not have been linked or formatted.

System Action: The procedure continues with RC=4.

User Response: Ensure that the access was not needed. If it was needed, you should make the minidisk accessible, and restart the procedure.

1906E **The access of *vdev* failed with a return code of *rc*. Processing stops for product *prodid*. [The original access order cannot be restored.]**

Explanation: VMFSETUP was not able to perform an access that was listed in the $PPF file. The minidisk might not have been linked or formatted. It has been determined that this minidisk might be essential for processing.

System Action: The procedure exits with RC=100.

User Response: Take whatever steps are needed to allow the minidisk to be accessed, and restart the procedure.

1907I **Assembling *filename* [(*options*)].**

Explanation: The assembly is going to begin. If you specified any assembler options, the options used are displayed.

System Action: The assembly begins.

User Response: None.

1908E **Error {assembling|updating} *filename*.**

Explanation: If assembling is taking place, an assembler error occurred. If updating is taking place, a severe update error occurred.

System Action: Execution is terminated.

User Response: Correct the error and rerun.

1909I *filename filetype* **A [has not been] created.**

Explanation: A text file with the given filename and filetype has been created, or the text file was not produced because of assembler errors.

System Action: None.

User Response: None.

1911E **Invalid CSECT name** *csect-name* **for ICS card in** *filename filetype.*

Explanation: The include control section card has an invalid CSECT name specified.

System Action: The procedure exits.

User Response: Correct the ICS card, and restart the procedure.

1912E **VER card missing in** *filename filetype.*

Explanation: A VER card was missing, and the patch cannot be performed.

System Action: The procedure exits with RC=100.

User Response: Add a VER card, and restart the procedure.

1932E **Part-handler** *execname* **for build list** *listname* **was found in the** *filename* **$PPF but a target was not specified.**

Explanation: The name of the target string for the specified part-handler and build list in the BLD section of the $PPF file is missing.

System Action: The procedure ends with RC=100.

User Response: Correct the $PPF file and reenter the command.

1937I **Merge processing has started for** *filename1 filename2.*

Explanation: The merge tag in the product parameter file contains at least one set of disk strings. It has been determined that the component being received has not been previously received and that at least one of the first disks of the disk strings were not empty.

System Action: The procedure continues with the merge of the disk strings. For each symbolic disk string listed on the merge tag, the contents of each disk is moved to the next, starting with the last pair in the symbolic disk string.

User Response: No response required.

1938I **The merge of** *symbolic-disk-string-name - dsk1* **to** *dsk2* **is complete.**

Explanation: As part of merge processing, files from each disk listed in a symbolic disk string are moved to the next disk starting with the last pair of disks. The contents of *dsk1* have been successfully copied to *dsk2* as a result of processing the symbolic disk string indicated in the message.

System Action: The procedure continues merging with the next pair of disks in the symbolic disk string. If no other pairs exist then if more symbolic disk strings exist on the merge tag, they are processed. Otherwise merge is complete.

User Response: No response required.

1939I **Merge processing is complete for component** *compname.*

Explanation: As part of merge processing, files from each disk listed in a symbolic disk string are moved to the next disk starting with the last pair of disks. This process is repeated for each symbolic disk string listed on the merge tag in the product parameter file. This message indicates that all symbolic disk strings have been merged.

System Action: The procedure continues receiving service for the component specified.

User Response: No response required.

1940I **Merge processing is not required for component** *compname.*

Explanation: The first disk of each symbolic disk string listed on the merge tag in the product parameter file has been found to be empty, making a merge unnecessary.

System Action: The procedure continues receiving service for the component specified.

User Response: No response required.

1941E **Component *compname* has already been received successfully from PUT|COR *tape-identifier*.**

Explanation: When a tape has successfully been received a merge indicator file is written to the first disk of each symbolic disk string listed on the merge tag in the product parameter file. This merge indicator has a filename of PUT or COR depending on tape type and a filetype of "$MR" concatenated to the first letter of the tape type - P or C, concatenated with the tape identifier. Another indicator file with a filename of "$" concatenated with the tape type and a filetype of the tape identifier is created with the comment in the COR or PUT DOCUMENT which contains a date and time stamp. The tape identifier from the "$MR" file is compared to the tape type currently being received. If both of these are identical then the comment in the second indicator file is compared with the comment from the current COR or PUT DOCUMENT. If both of these comments are identical then this message is issued.

System Action: The procedure exits with RC=100.

User Response: This message indicates that the service from this tape has already been successfully received. Check that the correct tape has been mounted. Also verify that the correct component has been entered. If the real intent is to re-receive service for this component then the indicator files described above can be renamed or erased so that it appears a receive has not been completed.

1942I **Service has been received for *partname parttype*. It is a part of the serviceability enhancements.**

Explanation: A Serviceability Enhancement part has received service. This service includes a new executable version for *partname*.

System Action: The procedure continues.

User Response: The executable part shipped for *partname* may be at a lower service level than your existing *partname*. Review the self-documenting information at the end of EXECs, XEDIT macros, and modules to determine the service levels of the new and existing parts. If necessary update the executable version used on your system to the latest level. Copy the PTF numbered version of the part to the executable version.

Example:

```
COPY VMFAPCOM EXC12345 B VMFAPCOM EXEC B
```

1943W **One or more AUX files have been received for manually supported parts. The AUX files for these parts must be evaluated and may need updating to assure the validity of your system.**

Explanation: The AUX files that have been received contain AUX entries for all updates to the part being serviced from the base level of the product to the update that was received. Some of the AUX entries in the new AUX file may already appear in other AUX files on your system. Having the same AUX entry appear more than once on your system can affect your system's integrity.

System Action: The procedure continues with RC=4.

User Response:

- Using the $VMFREC $HISTORY file that was created during execution of VMFREC EXEC, make a list of all of the AUX files that were received. The $VMFREC $HISTORY contains entries from all runs of VMFREC EXEC. The entries from the latest run are on the bottom and are outlined with beginning and ending time stamps.
- For each AUX file received, use the LISTFILE command to make a list of all of the existing AUX files for this part.
- Use XEDIT to view CNTRL file that is used for the given part.
- Identify the appropriate AUX level in the CNTRL file for the given part. For example, if you received corrective service and there is an AUXCOR level in the CNTRL file, then this would be the appropriate level for the new AUX file.
- If necessary, rename the new AUX file so that its filetype matches the appropriate AUX level in the CNTRL file.
- Compare the AUX entries in each AUX file level that is below the new AUX file level with the AUX entries in the new AUX file. For any duplicate entries, use an asterisk (*) to comment them out of the new AUX file.
- Compare the AUX entries in the old AUX file at the same level as the new AUX file with the new AUX file. If the old AUX file at this level contains more AUX entries, erase the new one or you will down level your system.

- Compare the AUX entries in each AUX file level that is above the new AUX file level. For any duplicate entries, use an asterisk (*) to comment them out of the higher level AUX files. If there are any Local Modifications or Local Text Patches, you must determine if they intersect with the new update and rework them if necessary.

1944E **Update file *filename1 filetype1 filemode1* contains an unsupported statement type *type* for the patch facility.**

Explanation: The update file identified in the message is a patch update which contains a statement other than NAME (Name for CSECT), ICS (Included control section), VER (verify), REP (replace) or comments. Only these statement types are allowed in patch update files.

System Action: The procedure continues with RC=100. Invalid cards are not included in the text deck.

User Response: While creating the update file, you may have incorrectly typed the update statements. If so, correct the statements and retry the procedure. Incorrect update files can generate a bad nucleus, therefore the update file should be corrected.

1946E **The *com* command failed with a return code *rc*. The input file specification was *filename1 filetype1 filemode1*. The output file specification was *filename2 filetype2 filemode2*.**

Explanation: The given CMS command experienced a severe error. If there is no input specification associated with the command, this part of the message is omitted. The same is true for the output specification.

System Action: The procedure continues with RC=100.

User Response: See the given CMS command for more information.

1947E **A 3 character abbreviation could not be found for the filetype *filetype* in the :APP. section of *prodid prodidft*.**

Explanation: Entries in the :APP. section of a $PPF (product parameter file) have filetypes and filetype abbreviations with which to rename the file that is being processed. If a file is being processed and its filetype abbreviation cannot be found, it is an error condition.

System Action: The procedure continues with RC=100.

User Response: Verify that you are using the correct level of the PPF and that it contains the required abbreviation in the :APP. section of the component that you are servicing.

1948W **AUX file *filename filetype* on *cuu* must be renamed by its filetype abbreviation and a PTF number, but it does not contain a PTF number. It is being renamed to *filename filetype2*.**

Explanation: AUX files that are supplied on service tapes have filetypes that may be listed in a control file. To avoid the new AUX file from being picked up and possibly conflicting with your system it must be renamed. The new filetype is the 3-character abbreviation for the AUX file which is found in the product parameter file (PPF) concatenated with the PTF number found in the top AUX entry in the AUX file. Since the AUX file does not contain any valid PTF numbers, the new filetype will be only the 3-character abbreviation.

System Action: The procedure continues with RC=4.

User Response: None.

1949W **PTF *ptfnum* for product *prodid* has already been applied.**

Explanation: A PTF was included in the apply list that had already been successfully applied. This determination was made by looking at the filemode of the $PTFPART file for this PTF. A filemode of 5 indicates that the PTF has been completely applied.

System Action: The procedure continues with RC=0.

User Response: None.

1953W *filename* **was interrupted the last time it was run.** *filename* **$ERRLOG has been restored from the saved history file** *filename* **$OLDLOG.**

Explanation: A copy of the $ERRLOG named $VMFREC, $VMFAPP, or $VMFBLD $OLDLOG is saved each time VMFREC, VMFAPPLY, or VMFBLD is invoked. If for some reason the execution of the EXEC is not completed (for example, if the user enters "HX" or the system abends) the $VMFxxx $OLDLOG remains on the users' A-disk.

Subsequent VMFREC, VMFAPPLY, or VMFBLD invocations append the $VMFxxx $OLDLOG to the new $VMFxxx ERRLOG and erase the $VMFxxx $OLDLOG. The $VMFxxx $OLDLOG will not exist after a complete run of the associated EXEC because it is erased, however the user should never erase the $VMFxxx $OLDLOG unless they do not want this historical exception log information.

System Action: This message has no effect on system action.

User Response: Investigate the previous messages to determine if further action is required.

1959E **The keyword ENV has been specified without any parameters following the keyword. Check the syntax and reenter the command.**

Explanation: The keyword ENV has been detected on the command line. This keyword requires the filename of an envelope file as a parameter and optionally a filetype and filemode. The required filemode was not found following the keyword.

System Action: The procedure exits with RC=8.

User Response: If you are attempting to process an envelope file, investigate the file specification of the file you want to process and reenter the command with the correct filename of the envelope. If you were not attempting to process an envelope file, reenter the command without the ENV keyword.

1960E **Envelope** *filename filetype filemode* **does not exist. Verify the file location or check the syntax and reenter the command.**

Explanation: The keyword ENV has been detected on the command line. If you specified an envelope filetype and filemode, a valid format envelope with the file specification given could not be found. If you did not specify a filemode, an envelope with the filename and filetype you specified could not be located on any accessed disk. If you only specified a filename, the default filetype of SERVLINK was used as the filetype and a valid envelope could not be located on an accessed disk.

System Action: The procedure exits with RC=28.

User Response: In order to process an envelope, the file must reside on an accessed disk at the start of the procedure. The envelope must also remain accessed during the full execution of the procedure. This requires that the envelope reside on the A-disk, or on a disk whose address is listed on a disk string in the product parameter file for the product being processed. If the disk is listed in the product parameter file, it must still be accessed at the start of the procedure. Verify the name and location of the envelope file you are attempting to process. Make sure that the disk access conditions described above are met. Reenter the command using the correct file specification with the envelope in the access order.

1961W **The Service Enhancement parts have been received from the {tape|envelope} header files but they have NOT been activated.**

Explanation: VMFREC has received the service execs from the first two tape or envelope files but due to a detected inconsistency they have not been renamed to their executable forms.

System Action: RC=4.

User Response: Examine other messages issued to determine the nature of the inconsistency, correct it and reissue VMFREC INFO.

1962W **No PTF numbered Service Enhancement parts were found in the {tape|envelope} header files.**

Explanation: VMFREC has determined that no Service Enhancement EXECs or related parts exist as PTF-numbered files in tape or envelope files one or two.

System Action: RC=4.

User Response: If PTF-numbered service EXECs were expected to be on the tape or in the envelope determine why they are missing. If the tape or envelope was created prior to having

PTF-numbered service enhancement parts in the header files then no action is required.

1963I **Level *nnnn* of the Service Enhancement parts is now established on *mode cuu*.**

Explanation: VMFREC has loaded the indicated level of the service EXECs onto the reported disk and renamed them so that they are executable.

System Action: RC=0.

User Response: None.

1964E **The update of the Service Enhancements part set inventory file $LEVEL EXEC was unsuccessful.**

Explanation: VMFREC was unable to write the $LEVEL EXEC file to disk while creating or updating it. This file contains a list of the service EXECs and associated parts that have been installed on the users system.

System Action: RC=100.

User Response: Ensure that the target disk [A|C] is writable and rerun VMFREC INFO.

1965E **The *name* command has failed with Return Code *rc* while operating upon file *filename filetype filemode cuu* execution is terminated.**

Explanation: The program was unable to execute the reported CMS command while operating on a CMS file.

System Action: RC=100.

User Response: Use the reported command name, return code and file specified to determine the appropriate action to take.

1966I **The *filename filetype filmode (cuu)* file saved as: *filename filetype filemode (cuu)*.**

Explanation: VMFREC has saved the existing service EXECs and associated parts prior to receiving a new version of these files.

System Action: RC=0.

User Response: None.

1967I **The Service Enhancements part set [on DISK *mode (cuu)* | in the {tape|envelope} header files] is at the *nnnn* level.**

Explanation: The level of the service EXECs and associated parts that is on the user's system is always reported. The level is extracted from the $LEVEL EXEC file. The service execs and associated parts are assumed to reside on the same disk as the $LEVEL EXEC file, and are also assumed to be at the level indicated in the $LEVEL EXEC file.

The level of the service EXECs and associated parts on the service tape or in the envelope is reported if it differs from the level the user has installed.

System Action: RC=0.

User Response: If the reported level differs from what is expected, determine why before proceeding. When the level on the tape or in the envelope is more current than the level on disk, examine the documentation supplied with the tape or envelope to determine if the new EXECs should be received via VMFREC INFO.

1968W **The Service Enhancements part set on DISK is at an UNKNOWN level**

or

The Service Enhancements part set in the {tape|envelope} header files is at an UNKNOWN level.

Explanation: Either VMFREC was unable to determine the level of the service EXECs on the user's system because it could not find a $LEVEL EXEC file, or VMFREC was unable to determine the level of the service EXECs on the service tape or envelope even though it contains some service enhancements parts in files one and two of the tape or envelope.

System Action: RC=4.

User Response: Determine why VMFREC is unable to find the $LEVEL EXEC file. Reestablish it manually or by running VMFREC INFO.
If the tape level is unknown, request a new service tape from IBM.

1969W The Service Enhancements part set in the tape header files, specified by $LEVEL $TAPE, conflicts with the tape inventory file $LEVEL MAP.

Explanation: VMFREC has found an inconsistency between the tape's inventory of service exec parts and the actual parts in tape files one and two. The tape is in error.

System Action: RC=4.

User Response: Request a new service tape from IBM.

1970W Entry for PTF *ptfno*, cannot be found in any AUX file for *filename*.

Explanation: VMFAPPLY is processing a list of all AUX file entries for the part identified in the message.

VMFAPPLY is attempting to verify that the PTF which contains service for this part exists in that list of AUX file entries but it cannot be found. (The PTF number would be in the third token of this list of AUX file entries.)

System Action: Processing continues.

User Response: This PTF is listed in the $APPLIST (apply list) but is not part of any AUX file for this part on your system. An AUX file on your system may be in error or the $APPLIST you are using may be in error. An AUX file could be in error because IBM shipped you an incorrect AUX file or you may have modified an AUX file that was sent to you from IBM.

Contact your IBM service representative for assistance.

1972W The service tape part inventory file $LEVEL MAP, is missing from the {tape|envelope}.

Explanation: VMFREC is unable to find a $LEVEL MAP file in tape file 1 of the service tape or envelope. Either the tape or envelope is in the old format or it is in error.

System Action: RC=4.

User Response: If the tape or envelope was expected to have PTF numbered service parts in files one and two then order a new service tape or envelope.

1973W The service {tape|envelope} may be in the old format. Using INFO (ALL may back level your Service EXECs).

Explanation: VMFREC has determined that the service tape or envelope may be in the old format and using the VMFREC INFO (ALL command may receive back level executable parts onto your disk.

System Action: RC=4.

User Response: Do not use VMFREC INFO (ALL unless you know it will not destroy your production service EXECs.

1974I A $PPF file for product *prodid* exists on this {tape|disk} volume but is not installed on your system. It must be received prior to servicing that product.

Explanation: VMFREC has checked the $PPF files in tape file two of the service tape or envelope and has determined that a $PPF file exists on the tape or envelope that is not listed in the $LEVEL EXEC file.

System Action: RC=0.

User Response: Before servicing the indicated product ensure that its new $PPF has been installed on your system.

1975W An UNKNOWN level of the Service Enhancement parts are now established on *mode cuu*.

Explanation: VMFREC INFO (ALL was used in a situation that prevents VMFREC from knowing the level of the service EXECs and other parts received from files 1 and 2 of the service tape.

System Action: RC=4.

User Response: Before performing additional service, ensure that the new service EXECs are those you wish to have installed.

Summary of Changes

Fourth Edition

Form of Publication: SC23-0364-3

Level of Product: VM/XA System Product Release 2.1

Date of Publication: June 1990

Changes to this publication:

- **New device support**

  VM/XA System Product Release 2.1 supports the 3390-1 and 3390-2 DASD devices. The 3390 Starter System can be used with a 3390-1 and 3390-2. New sample directories and minidisk maps for both of the 3390 models have been added to Appendix C, "VM/XA System Product Starter System Information."

- **Single-level AUX**

  Single-level AUX fulfills three customer requirements:

  - The AUX structure has been significantly simplified
  - The LOCAL string is no longer used for nonlocal service
  - The need to reconcile your VM/SUP reach-ahead and COR service when installing a PUT has been removed

  You can still use the multiple AUX file scheme if you choose to.

- **ServiceLink**

  ServiceLink is a part of IBMLink that lets you access IBM software service information online. You must be authorized to use IBMLink and ServiceLink. In addition, you must be authorized to order service and receive it electronically using ServiceLink.

  The ServiceLink information in this manual is intended to help you use the VM support that allows you to receive service through a disk file, in addition to the conventional tape medium.

- **Receiving multiple COR tapes**

  A procedure has been added to allow you to receive multiple COR tapes before running VMFAPPLY. To use this procedure, you need to have APARs VM40518 and VM41018 installed on your system. You will find this procedure in the following chapters:

  - Chapter 10, "Receiving Service for CMS" on page 417
  - Chapter 11, "Receiving Service for CP" on page 423
  - Chapter 16, "Service to DV" on page 507
  - Chapter 17, "Service to GCS" on page 517

- **APAR VM37518**

  The information supporting APAR VM37518, contained in *VM/XA SP Serviceability Enhancements Program Update Information APAR VM37518*, GC23-0503-0, has been included in this edition. Support provided by APAR 37518 includes:

  - Additional parts support provided by part handlers VMFRCAUX and VMFAPAUX.
  - An exception log viewing tool called VMFVIEW EXEC.
  - A reduction in the number of on-screen messages from VMFAPPLY EXEC.
  - A TASK minidisk string, accessed after the A-disk and before the component database.

- Read/only minidisk specification in the product parameter file.
- A check for the existence of $PTFPART lists and PTF-numbered parts on the database by part handlers VMFRCCOM and VMFRCUPP, so that these parts are not reloaded if they already exist.
- Inprovements in VMFAPPLY performance using filemode 5 on the PTF parts list to indicate that the PTF has been succesfully applied. Additional control over this process is given in the product parameter file by the CHKUPD tag.

- **Procedure changes.**

  Many of the procedure steps in the installation and service parts of this manual have been re-written to improve usability.

Glossary

A

APAR. Authorized program analysis report.

authorized problem analysis report (APAR). (1) A report of a specific system problem. (2) The code correcting a specific system problem.

automatic software re-IPL. The process by which the control program attempts to restart the system after abnormal termination. This process does not involve the hardware IPL process. See also *virtual=real machine recovery*.

C

CCS. Console communication services.

CCW. Channel command word.

channel command word (CCW). A doubleword structure that directs an I/O operation on a device or channel and includes pointers to any storage areas associated with the operation. One or more CCWs make up a channel program.

circumventive fix. A fix that branches around the code in error, or avoids executing that code in some other way, rather than correcting the error. A circumventive fix is temporary.

CMS. Conversational monitor system.

control program (CP). The component of VM/XA SP that manages the resources of a single System/370-Extended Architecture system so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of either a System/370 computing system or a System/370-Extended Architecture computing system.

console communication services (CCS). A group of CP routines that interface with the VTAM service machine, providing full VM/XA SP console capabilities for SNA/CCS terminal users.

conversational monitor system (CMS). The component of VM/XA SP that, as a virtual machine operating system, provides interactive time-sharing. CMS allows users to communicate with the system and with each other, to create and edit files, and to develop and run application programs. It operates in either System/370 mode or 370-XA mode under the control of CP.

COR. Corrective service tape.

COR-closed PTF. A PTF that is available on a corrective service tape.

CP. Control program.

corrective service. Service sent to you by IBM to correct a specific problem, supplied on a corrective service tape in the same format as a program update tape.

corrective service tape (COR). A tape containing customized corrective service, sent to you by IBM.

D

DCSS. Discontiguous saved segment.

directory. A CP disk file that includes an entry for each user in the system. The entry defines the characteristics of the user's initial virtual machine configuration. These characteristics include the user ID, the password, normal and maximum allowable virtual storage, virtual device definitions, the privilege class, the dispatching priority, logical line editing characters, and the account number.

discontiguous saved segment (DCSS). A saved segment that occupies one or more architecturally-defined segments. It begins and ends on segment boundaries. It is accessed by its own name. Contrast with *member saved segment*. See also *saved segment, segment,* and *segment space*.

disk string. See string.

dump viewing facility. A VM/XA SP component that allows users to display, format, and print data interactively from CP hard and soft abend, stand-alone, and virtual machine dumps, and to process CP trace table data stored on tape or in a system trace file.

dynamic paging area. The area of real storage allocated by CP for V=V machine paging. This area also contains CP nonresident modules, CP control blocks, CP trace tables, free storage pages, and the alternate processor's prefix storage areas.

E

Expanded Storage. Optional integrated high-speed storage. In VM/XA SP, Expanded Storage may be shared by CP and one or more virtual machines. It may also be dedicated to CP or to a particular virtual machine.

F

full-pack minidisk. A virtual disk that contains all of the addressable cylinders of a real DASD volume.

full-screen mode. In VM/XA SP, the environment in which an entire 3270 display screen is under the control of a program running in a virtual machine.

G

GCS. group control system

group control system (GCS). The component of VM/XA SP that, as a virtual machine supervisor, executes in a group of System/370 virtual machines under CP control to provide an interface that helps support a native Systems Network Architecture (SNA) network.

guest. An operating system running in a virtual machine managed by the VM/XA SP control program. Contrast with *host.*

guest real storage. The storage that appears real to the operating system running in a virtual machine. Contrast with *guest virtual storage, host real storage,* and *host virtual storage.*

guest virtual storage. The storage that appears virtual to the operating system running in a virtual machine. Contrast with *guest real storage, host real storage,* and *host virtual storage.*

H

host. The VM/XA SP control program in its capacity as manager of a virtual machine in which another operating system is running. Contrast with *guest.*

host real storage. The storage that appears real to the control program. If VM/XA SP is running native, this is real storage; if VM/XA SP is running in a virtual machine, this is virtual storage. Contrast with *guest real storage, guest virtual storage,* and *host virtual storage.*

host virtual storage. The storage that appears virtual to the control program. Contrast with *guest real storage, guest virtual storage,* and *host real storage.*

I

image library. A set of modules, contained in a system data file, that define the spacing, characters, and copy modification data that a 3800 printer uses to print a spool file or that define the spacing and character set that an impact printer uses to print a spool file. See also *system data file.*

inter-user communication vehicle (IUCV). A generalized CP interface that facilitates the transfer of data among virtual machines.

IUCV. Inter-user communication vehicle.

L

local service. Any service not supplied on a PUT or COR tape. It can be service that you originate, or service that is sent to you by IBM to correct or circumvent a specific problem that you have encountered and reported. Local service can be supplied on a tape, in hard copy, or over the phone.

M

member saved segment. A saved segment that begins and ends on a page boundary. It belongs to from 1 to 64 segment spaces and is accessed either by the segment space name or its own name. Contrast with discontiguous saved segment. See also *saved segment, segment,* and *segment space.*

message repository file. A type of system data file that contains a set of VM/XA SP messages translated into a national language.

minidisk string. See string.

missing interrupt handler. A CP function for detecting and dealing with real I/O operations that do not complete within a specified time.

multiple preferred guests. A VM/XA SP facility that supports up to six preferred virtual machines when the Processor Resource/Systems Manager* (PR/SM* feature is installed in the real machine. See also *preferred virtual machine.*

Processor Resource/Systems Manager and PR/SM are trademarks of the IBM Corporation.

N

named saved system (NSS). A copy of an operating system that a user has named and retained in a system data file. The user can load the operating system by its name, which is more efficient than loading it by device number. See also *discontiguous saved segment, member saved segment, saved segment, segment space,* and *system data file.*

NSS. Named saved system.

O

object-maintained code. Code serviced by replacement of a complete part. See also *replacement service,* and *source-maintained code.*

P

pageable virtual machine. Synonymous with *virtual=virtual machine.*

preferred virtual machine. A virtual machine that runs in the V=R area. CP gives this virtual machine preferred treatment in the areas of performance, processor assignment, and I/O interrupt handling. See also *multiple preferred guests, virtual=fixed machine, virtual=real area,* and *virtual=real machine.*

preventive service. Program update service.

Processor Resource/Systems Manager (PR/SM). A separately orderable feature available with 3090E processors that provides for logical partitioning of the real machine and support of multiple preferred guests. See also *multiple preferred guests.*

PR/SM. Processor Resource/Systems Manager.

program temporary fix (PTF). A package of one or more APARs.

program update service. Service sent to you on a program update tape, also called preventive service. Not to be confused with update service.

program update tape (PUT). A tape containing customized service. Each service tape contains cumulative service for the customer's products back to the earliest release level of the product still supported.

PTF. Program temporary fix.

PUT. Program update tape.

R

real system operator. Any user who loads and runs VM/XA SP in the real machine. Contrast with *virtual machine operator.*

remedial service. Service to temporarily correct or circumvent a specific problem that you have encountered and reported, until a PTF becomes available. Remedial service includes corrective service, which IBM supplies on a corrective service tape with the same format as a program update tape, and local service, which IBM supplies in some other format or which you originate yourself.

replacement service. Service by replacement of the entire part that is in error. See also *object-maintained code,* and *update service.*

S

saved segment. One or more pages of storage that have been named and retained in a system data file. See also *discontiguous saved segment, member saved segment, segment, segment space,* and *system data file.*

segment. In System/370 architecture, 64 kilobytes of storage. In 370-XA architecture, 1 megabyte of storage. See also *saved segment.*

segment space. A saved segment composed of up to 64 member saved segments accessed by a single name. A segment space occupies one or more architecturally-defined segments; it begins and ends on segment boundaries. A user with access to a segment space has access to all of its members. See also *discontiguous saved segment, member saved segment, saved segment,* and *segment.*

service tape. Program update tape or corrective service tape.

service virtual machine. A virtual machine that provides system services. These services include accounting, error recording, monitoring, and those provided by supported licensed programs.

SMSG function. A CP function that allows a virtual machine to send a special message to another virtual machine programmed to accept and process the message. See also *special message.*

SNA. Systems Network Architecture

SNA/CCS terminal. Any terminal accessing VM/XA SP that is managed by a VTAM service machine.

source-maintained code. Code serviced by combining updates with the base code. See also *object-maintained code* and *update service.*

special message. A data transmission, made up of instructions or commands, sent from one virtual machine to another via the SMSG function. A special message is processed by the receiving virtual machine and does not appear on the receiver's console. See also *SMSG function.*

spool file. A collection of data along with CCWs for processing on a unit record device. Contrast with *system data file.*

string. In applying service, a set of minidisks defined in the product parameter file for a particular use; for example, the BASE2 string, which holds source code, or the DELTA string, which holds update files from the program update tape.

SVC 76. In VM/XA SP, a supervisor call instruction that records the error incidents encountered by certain operating systems running in virtual machines. When a virtual machine operating system issues an SVC 76, VM/XA SP translates the virtual storage and I/O device addresses to real addresses, records the information on the VM/XA SP error recording virtual machine, and returns control to the issuing virtual machine. This interface bypasses the virtual machine's own error recording routine, and avoids duplicate error recording.

System/370 mode. A virtual machine operating mode in which System/370 functions are simulated. Contrast with *370-XA mode.*

system data file. A collection of data associated with a particular function. Types of system data files include saved segments, NSSs, UCR files, image libraries, message repository files, and system trace files. Because a system data file contains no CCWs, it cannot be processed on a unit record device. Contrast with *spool file.*

system hold status. A spool file status that prevents a file from being printed, punched, or read until the real system operator releases it. Contrast with *user hold status.*

system trace file. A type of system data file that contains CP or virtual machine trace data.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration of, networks.

U

UCR file. User class restructure file.

unit record device. A reader, a printer, or a punch.

update service. Service by applying updates to the base code (also called source code) that is in error. Not to be confused with program update service. See also *replacement service,* and *source-maintained code.*

user class restructure file (UCR file). A type of system data file that contains information used to override the IBM-defined privilege class structure of CP commands, DIAGNOSE instruction codes, and certain CP system functions.

user directory. See directory.

user hold status. A spool file status that prevents a file from being printed, punched, or read until the file owner releases it. Contrast with *system hold status.*

V

Vector Facility (VF). A hardware feature that provides synchronous instruction processing for high-speed manipulation of fixed-point and floating-point data.

VF. Vector Facility.

V=F machine. Virtual=fixed machine.

virtual=fixed machine (V=F machine). A preferred virtual machine with a fixed, contiguous area of host real storage that does not start at page 0. CP provides performance enhancements for this virtual machine. See also *multiple preferred guests, preferred virtual machine, virtual=real area, virtual=real machine,* and *virtual=virtual machine.*

virtual machine. In VM/XA SP, a functional equivalent of either a System/370 computing system or a System/370-Extended Architecture computing system. Each virtual machine is controlled by an operating system. VM/XA SP controls the concurrent execution of multiple virtual machines on an actual System/370-Extended Architecture system.

Virtual Machine/Extended Architecture System Product (VM/XA SP). An operating system that allows multiple IBM System/370 and 370-XA operating systems to run simultaneously on a single 370-XA processor. The multiple systems may be used for production, testing, developing application programs, maintenance, and migration. VM/XA SP also provides a high-capacity interactive environment. There are four components: the control program (CP), the conversational monitor system (CMS), the dump viewing facility, and the group control system (GCS).

virtual machine operator. Any user who loads and runs an operating system in a virtual machine. Contrast with *real system operator.*

virtual=real area (V=R area). A fixed, contiguous section of real storage, starting at page 0, in which preferred virtual machines execute. CP does not page

this storage. See also *preferred virtual machine, virtual=fixed machine,* and *virtual=real machine.*

virtual=real machine (V=R machine). A preferred virtual machine with a fixed, contiguous area of host real storage that starts at page 0. CP provides performance enhancements and an automatic recovery facility for this virtual machine. See also *multiple preferred guests, preferred virtual machine, virtual=real area, virtual=real machine recovery,* and *virtual=virtual machine.*

virtual=real machine recovery (V=R machine recovery). A CP function that allows the V=R machine to resume operation after most CP abnormal terminations. When possible, the facility reestablishes the V=R machine environment, allowing the operating system running in that virtual machine to perform its own recovery processes. See also *automatic software re-IPL.*

virtual=virtual machine (V=V machine). A virtual machine that runs in the dynamic paging area. CP pages this virtual machine's guest real storage in and out of host real storage. See also *dynamic paging area, virtual=fixed machine,* and *virtual=real machine.*

virtual supervisor state. A condition, controlled by a virtual machine's current PSW, during which the control program allows the virtual machine to issue input/output and other privileged instructions. When these instructions are not emulated, the control program intercepts these instructions and simulates their functions for the virtual machine.

virtual wait time. The period during which the control program suspends the processing of a program while a required resource is unavailable.

VM/XA SP. Virtual Machine/Extended Architecture System Product.

VTAM service machine. A collection of networking programs running in a virtual machine that, together with the CP console communication services (CCS) routines, provide full VM/XA SP console capabilities for SNA/CCS terminal users. A VTAM service machine contains either (1) VM/VTAM with VSCS running as an application under control of GCS, or (2) VM/VCNA running as a VTAM application under control of the VSE or VS1 operating system.

V=R area. Virtual=real area.

V=R machine. Virtual=real machine.

V=R machine recovery. Virtual=real machine recovery.

V=V machine. Virtual=virtual machine.

Numerics

370 mode. Synonym for System/370 mode.

370-XA mode. A virtual machine operating mode in which System/370-Extended Architecture functions are simulated. Contrast with *System/370 mode.*

Bibliography

This bibliography lists the names and order numbers of microfiche and publications on VM/XA System Product.

VM/XA System Product Microfiche

You can order microfiche listings that contain code. The order numbers for the microfiche are:

Order No.	Description
LYC7-0347	VM/XA System Product: CP listings
LYC7-0348	VM/XA System Product: CMS listings
LYC7-0349	VM/XA System Product: GCS listings
LYC7-0350	VM/XA System Product: dump viewing facility listings.

VM/XA System Product Publications

The publications are shown in Figure 69 on page 910. You can order them by their individual order numbers, or you can order **most** of them as a group by using a single order number, SBOF-0241. SBOF-0241 provides:

- All unlicensed publications (order numbers that do not begin with LY)
- Enough three-ring binders to hold the publications
- Spine and cover inserts for the binders.

Note that you must order manuals that contain licensed information (manuals with order numbers that begin with LY) through your support personnel. Books that contain licensed information are:

- *VM/XA System Product: Features Summary*, LY27-8058
- *VM/XA System Product: Diagnosis Guide*, LY27-8056
- *VM/XA System Product: CP Diagnosis Reference*, LY27-8054
- *VM/XA System Product: CMS Diagnosis Reference*, LY27-8052
- *VM/XA System Product: Group Control System Diagnosis Reference*, LY27-8060
- *VM/XA System Product: CP Data Areas and Control Blocks*, LY27-8053
- *VM/XA System Product: CMS Data Areas and Control Blocks*, LY27-8051.

As shown in Figure 69 on page 910, VM/XA SP publications are organized into six categories:

1. **Evaluation and introduction:** information on VM/XA SP concepts
2. **Planning, installation, administration, and service:** planning your system and performing system installation and maintenance
3. **Operation and end use:** performing system and virtual machine tasks
4. **Application programming:** information on using programming interfaces
5. **Diagnosis:** information for understanding of VM/XA SP design and to aid in problem diagnosis
6. **Reference:** quick retrieving of library usage information, command language syntax, macro instructions, DIAGNOSE codes, and system messages.

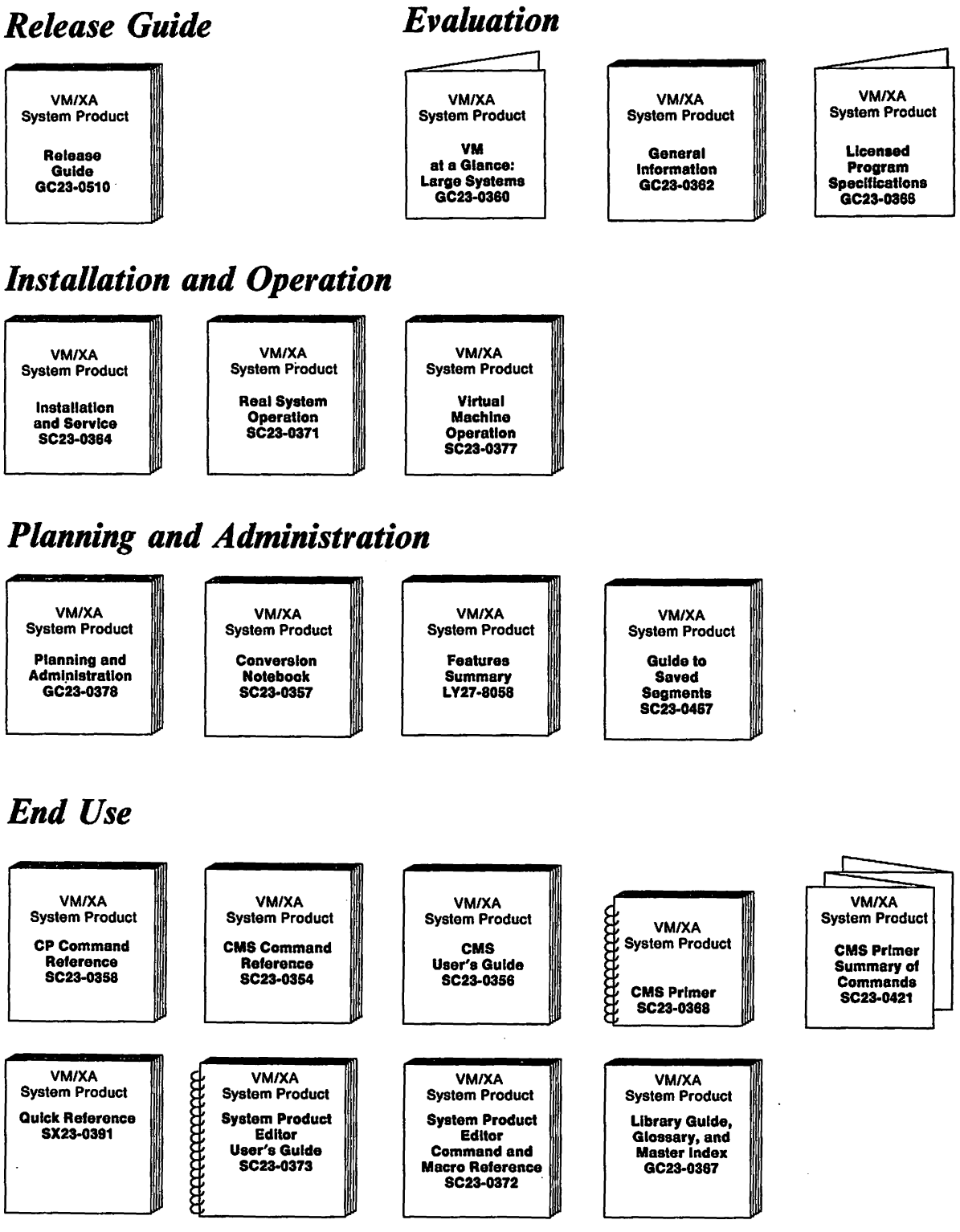
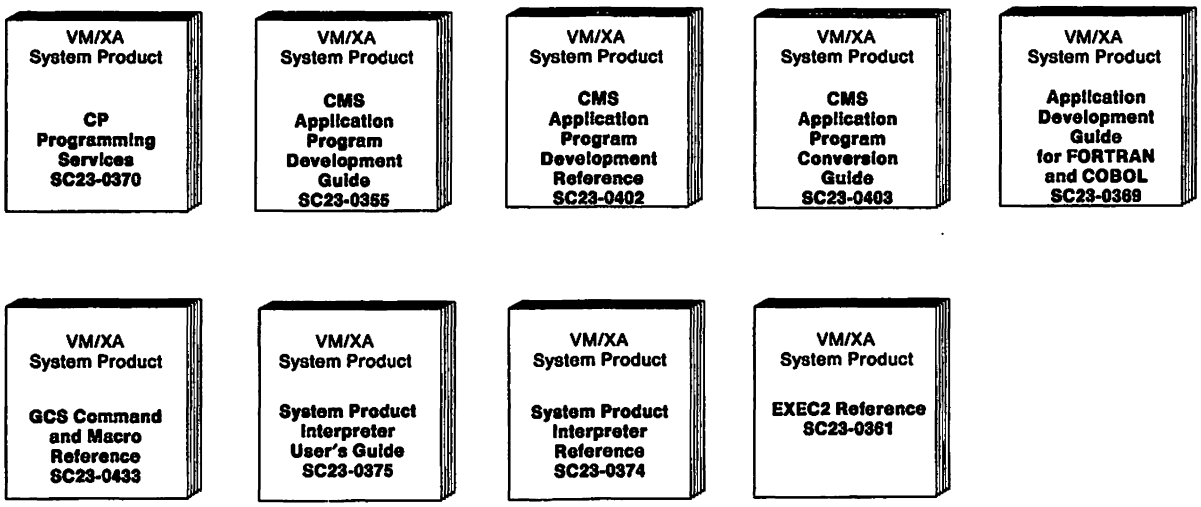
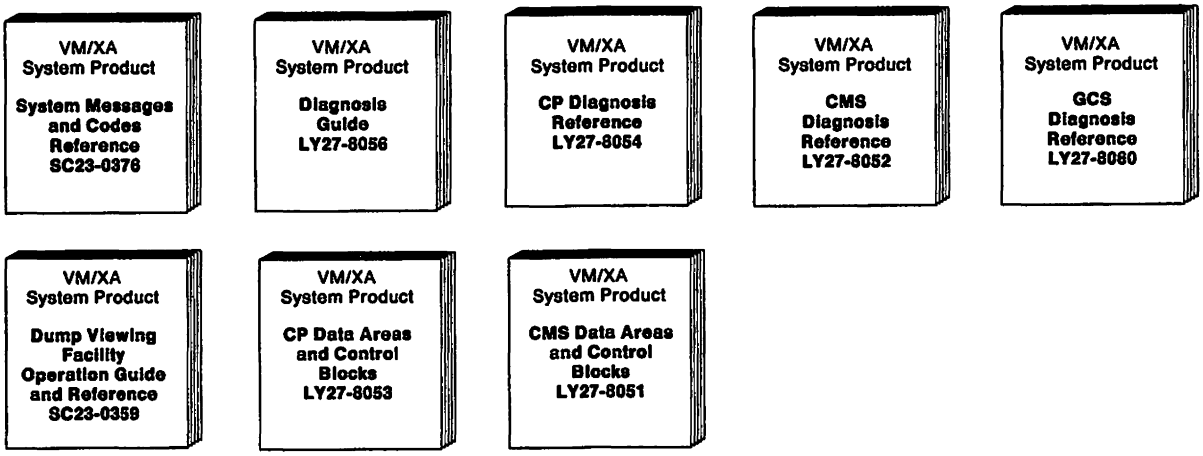


Figure 69 (Part 1 of 2). VM/XA System Product Publications

Application Programming



Diagnosis



Binders and Inserts

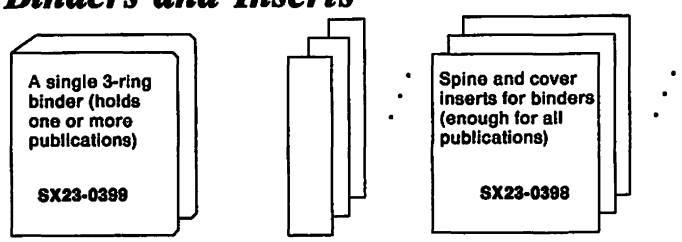


Figure 69 (Part 2 of 2). VM/XA System Product Publications

Evaluation and Introduction: Understanding Basic System Concepts

The evaluation and introduction publications for VM/XA SP are:

- *VM/XA System Product: Licensed Program Specifications*, GC23-0366

  Provides information on the warranted functions of VM/XA SP and describes the specified operating environment.

- *VM at a Glance: Large Systems*, GC23-0360

  Presents an overview of the features of each of the large VM systems: the VM/XA Systems Facility, VM/SP High Performance Option, and VM/XA System Product.

- *VM/XA System Product: General Information*, GC23-0362

  Provides general and planning information for VM/XA SP. It can help you decide whether VM/XA SP can fill your needs.

- *VM/XA System Product: Conversion Notebook*, SC23-0357.

  Provides migration and compatibility information for customers migrating from VM/SP HPO Release 5 and VM/XA SF Release 2.

Planning, Installation, Service, and Administration: Generating and Maintaining the System

- *VM/XA System Product: Planning and Administration*, GC23-0378

  Presents system planning concepts for VM/XA SP and virtual machine planning concepts for guest operating systems. Planning topics include suggestions for defining your real system configuration and building and updating your directory. This book discusses running these operating systems under VM/XA SP: MVS/SP, MVS/XA, VSE/SP, VM/SP, and VM/SP HPO.

  This book also provides information on how to manage your system. Administration topics include:

  - Setting up virtual machines for accounting, error recording, and CMS batch
  - Setting up the programmable operator
  - Redefining command privilege classes
  - Defining and managing saved segments and named saved systems
  - Tuning the system
  - Reference information on the VM/XA SP monitor

- *VM/XA System Product: Installation and Service*, SC23-0364.

  Gives step-by-step procedures for generating VM/XA SP and describes how to apply service updates to your system.

Operations and End Use: Making the System Work for You

- *VM/XA System Product: Real System Operation*, SC23-0371

  Provides a task-oriented source for real system operations. In step-by-step format it describes the procedures and commands used to perform each real system task.

- *VM/XA System Product: Virtual Machine Operation*, SC23-0377

  Provides a task-oriented source for virtual machine operations. In step-by-step format it describes the procedures and commands used to perform each virtual machine task.

- *VM/XA System Product: CMS User's Guide*, SC23-0356

  Provides information on using CMS.

- *VM/XA System Product: CMS Primer*, SC23-0368

  Provides a tutorial approach to learning CMS.

- *VM/XA System Product: System Product Editor User's Guide*, SC23-0373.

  Provides information about using the System Product Editor.

Application Programming: Using Programming Interfaces

- *VM/XA System Product: CP Programming Services*, SC23-0370

  Provides reference and usage information for the following CP services and macros:

  - The DIAGNOSE codes
  - The IUCV macro
  - CP system services

- *VM/XA System Product: CMS Application Program Development Guide*, SC23-0355

  Helps you use the assembler language macros and functions of CMS in your assembler language application programs. It describes how to manage storage, perform I/O, handle interrupts, process abends, load and start programs, and exploit 31-bit addressing. It also includes message repository information.

- *VM/XA System Product: CMS Application Program Conversion Guide*, SC23-0403

  Helps you convert your existing CMS assembler language application programs so that they run on the CMS provided with VM/XA SP. It summarizes the differences between the CMS provided with VM/XA SP and previous versions of CMS, it describes the tasks you may need to perform in converting your programs, and it points you towards other books that can help you convert your programs.

- *VM/XA System Product: Application Development Guide for FORTRAN and COBOL*, SC23-0369

  Provides information on how to use the CMS environment to develop and execute FORTRAN and COBOL application programs. The book contains such information as:

  - How to use the System Product Editor to create an application program
  - How to load, compile, and execute selected supported licensed programs

- *VM/XA System Product: System Product Interpreter User's Guide*, SC23-0375.

  Provides information about using the System Product Interpreter.

Diagnosis: Understanding System Design

- *VM/XA System Product: Diagnosis Guide*, LY27-8056

  Provides diagnostic information. It describes how to locate problems within the VM/XA SP control program, and how to describe and report problems to IBM support personnel. The diagnosis reference publications describe how the system works. You should use them as supplements to this book.

- *VM/XA System Product: CP Diagnosis Reference*, LY27-8054

  Describes each of the major VM/XA SP control program facilities. Also contains a module cross-reference list.

- *VM/XA System Product: CMS Diagnosis Reference*, LY27-8052

  Describes each of the major conversational monitor system facilities.

- *VM/XA System Product: Group Control System Diagnosis Reference*, LY27-8060

  Describes step-by-step procedures for identifying problems in each of the major facilities of the group control system and lists important GCS control blocks.

- *VM/XA System Product: CP Data Areas and Control Blocks*, LY27-8053

  Lists the data areas and control blocks used by the VM/XA SP control program.

- *VM/XA System Product: CMS Data Areas and Control Blocks*, LY27-8051

  Lists the data areas and control blocks used by CMS.

- *VM/XA System Product: Dump Viewing Facility Operation Guide and Reference*, SC23-0359.

  Describes step-by-step procedures for using the dump viewing facility. The publication is also a reference for dump viewing facility commands and messages.

Reference: Retrieving Information Quickly

- *VM/XA System Product: CP Command Reference*, SC23-0358

  Provides complete descriptions of the commands used to communicate with VM/XA SP, including usage notes. The commands are in alphabetical order.

- *VM/XA System Product: CMS Command Reference*, SC23-0354

  Provides complete descriptions of the commands used to communicate with the CMS component of VM/XA SP. The commands are in alphabetical order.

- *VM/XA System Product: Group Control System Command and Macro Reference*, SC23-0433

  Provides complete descriptions of the commands used to communicate with the GCS component of VM/XA SP and the macros that can be used in GCS application programs. This book also includes information about writing REXX EXECs for use under GCS.

- *VM/XA System Product: Features Summary*, LY27-8058

  Provides a comprehensive survey of VM/XA SP at a higher level than the *VM/XA SP CP Diagnosis Reference*. Topics cover such areas as:

  - Supported features, hardware, and operating systems
  - CP-owned direct access storage, CP virtual storage, and real storage organization
  - CP virtual and real machine management
  - CMS, GCS, and the dump viewing facility
  - VM/XA SP performance facilities and installation

- *VM/XA System Product: System Messages and Codes Reference*, SC23-0376

  Contains the system messages generated by the VM/XA SP CP, CMS, and GCS components. For each message, the publication provides:

  - The message number
  - The message text
  - An explanation of why the message was issued
  - System action
  - Recommended operator action (if any)
  - Recommended user action (if any)
  - Return code (if any)

  The publication also documents all abend codes and wait state codes, as well as the reason for each code and the recommended action.

- *VM/XA System Product: CMS Application Program Development Reference*, SC23-0402

  Describes the CMS programming interface. It includes descriptions of the CMS macros, DOS macros, and external-use control blocks.

- *VM/XA System Product: Quick Reference*, SX23-0391

  Shows only the command syntax of all the VM/XA SP commands. The commands summarized in this publication are described in detail in the *VM/XA System Product: CP Command Reference*, the *VM/XA System Product: CMS Command Reference*, the *VM/XA System Product: Dump Viewing Facility Operation Guide and Reference*, and the *VM/XA System Product: Group Control System Command and Macro Reference*.

- *VM/XA System Product: System Product Editor Command and Macro Reference*, SC23-0372

  Describes the system product editor commands, in alphabetical order.

- *VM/XA System Product: System Product Interpreter Reference*, SC23-0374

  Describes the system product interpreter statements, in alphabetical order.

- *VM/XA SP EXEC 2 Reference*, SC23-0361

  Describes the EXEC 2 control words, in alphabetical order.

- *VM/XA System Product: Library Guide, Glossary, and Master Index*, GC23-0367

  Provides an overview of the library's structure, a glossary, and a means for directly locating specific information within a manual or manuals.

- *VM/XA System Product: CMS Primer Summary of Commands*, SC23-0421.

  Contains summary information about commands described in the *VM/XA System Product: CMS Primer*.

Index

D

E

F

G

H

M

N

O

P

R

S

T

U

V

W

X

Z

Numerics

Special Characters

Virtual Machine/
Extended Architecture
System Product
Release 2.1

SYSTEM
USABILITY
COMMENTS

Please use this form to communicate your comments about the usability of the VM system, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the Product Usability Department for appropriate review and action, if any. Comments may be written in your own language; English is not required.

System Information

If you answer **No**, please explain.

	Yes	No
• Does the VM system meet your needs?	☐	☐
• Is it easy to use and understand?	☐	☐
• Are the commands/messages easy to understand and use?	☐	☐
• Are the HELP facilities appropriate?	☐	☐

Customer Information

- What is your occupation? _____
- How long have you been in this occupation? _____
- How long have you been using VM? _____
- Indicate the tasks your job involves:

Evaluation	☐	Planning	☐
Installation	☐	Administration	☐
Customization	☐	Operations	☐
Diagnosis	☐	End Use	☐
Other	☐		

Your Comments:

We appreciate your comments.

If you would like a reply, please supply your name and address on the reverse side of this form. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Note: Staples can cause problems with automatic mail-sorting equipment.
Please use pressure-sensitive or other gummed tape to seal this form.

System Usability Comments

Fold and Tape | Please Do Not Staple | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department 635
P.O. Box 6, Route 17C Glendale Drive
Endicott, New York 13760

Fold and Tape | Please Do Not Staple | Fold and Tape

If you would like a reply, *please print:*

Your Name _____

Company Name _____ *Department* _____

Street Address _____

City _____

State _____ *Zip Code* _____

IBM Branch Office serving you _____

IBM®

PRINTED IN U.S.A.

Reader's Comments

Virtual Machine/
Extended Architecture
System Product
Installation and Service
Release 2.1

Publication No. SC23-0364-3

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ]	As an introduction	[ ]	As a text (student)
[ ]	As a reference manual	[ ]	As a text (instructor)
[ ]	For another purpose (explain)		

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

Name

Address

Company or Organization

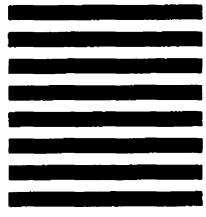
Phone No.

Reader's Comments
SC23-0364-3

Cut or Fold
Along Line

Fold and Tape | Please do not staple | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department 52QA MS 511
NEIGHBORHOOD ROAD
KINGSTON NY 12401-0000

Fold and Tape | Please do not staple | Fold and Tape

Cut or Fold
Along Line

Program Number
5664-308

File Number
S370-34

SC23-0364-3

Printed in U.S.A.